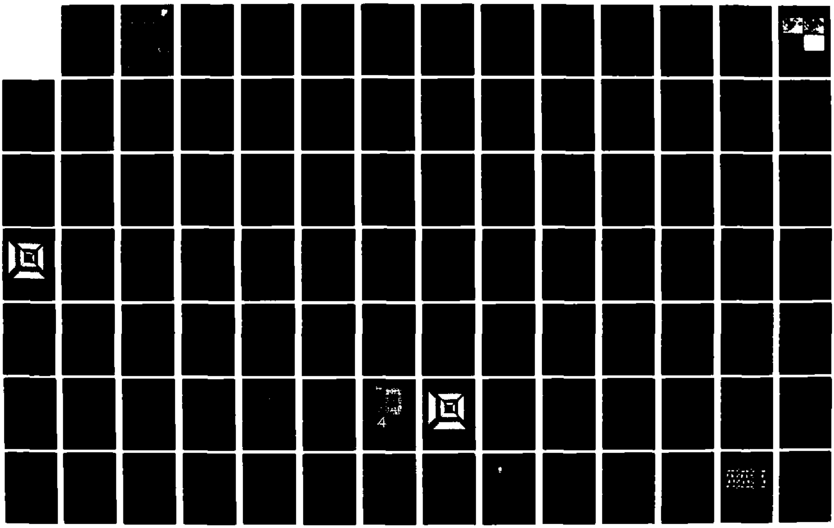
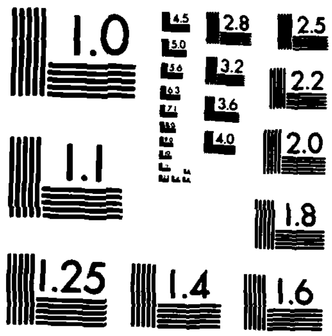
AD-A120 982 TRANSPORTABLE MAPS SOFTWARE VOLUME I(U) CONTROL DATA CORP MINNEAPOLIS MN INFORMATION SCIENCES DIV A E LABONTE JUL 82 9199500 RADC-TR-82-194 1/4

UNCLASSIFIED F30602-80-C-0326 F/G 20/6 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD A120982

(12)

RADC-TR-82-194
Final Technical Report
July 1982

TRANSPORTABLE MAPS SOFTWARE

Control Data Corporation

A. E. LaBonte

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, NY 13441

This report has been reviewed by the RADC Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RADC-TR-82-194 has been reviewed and is approved for publication.

APPROVED: John T. Boland

JOHN T. BOLAND
Project Engineer

APPROVED:

A. J. DRISCOLL, Colonel, USAF
Chief, Intelligence & Reconnaissance Division

FOR THE COMMANDER: John P. Huss

JOHN P. HUSS
Acting Chief, Plans Office

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (IRRA) Griffiss AFB NY 13441. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document requires that it be returned.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER RADC-TR-82-194	2. GOVT ACCESSION NO. AD-A120 982	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) TRANSPORTABLE MAPS SOFTWARE		5. TYPE OF REPORT & PERIOD COVERED Final Technical Report
		6. PERFORMING ORG. REPORT NUMBER 9199500
7. AUTHOR(s) A. E. LaBonte		8. CONTRACT OR GRANT NUMBER(s) F30602-80-C-0326
9. PERFORMING ORGANIZATION NAME AND ADDRESS Control Data Corporation Information Sciences Division Minneapolis MN 55440		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 62702F 45941831
11. CONTROLLING OFFICE NAME AND ADDRESS Rome Air Development Center (IRRA) Griffiss AFB NY 13441		12. REPORT DATE July 1982
		13. NUMBER OF PAGES 332
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Same		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Same

18. SUPPLEMENTARY NOTES

RADC Project Engineer: John T. Boland (IRRA)

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Image Compression
Image Processing
Intelligence

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report covers the development of a transportable software package in FORTRAN for the MAPS image compression technique. Also included are a user's manual for the developed software, a software maintenance manual and the software listing. Possession of the report will allow a user to install and operate this software on DEC PDP 11/45, 11/70 and VAX 11/780 processors.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

PREFACE

This is Volume I of the Final Technical Report on Transportable MAPS Software. It describes the development of the TransMAPS software package; its companion volumes contain the TransMAPS User's Manual (Volume II) and TransMAPS Maintenance Manual (Volume III). This volume is submitted in fulfillment of CDRL item A002 of Contract # F30602-80-C-0326.

TABLE OF CONTENTS

VOLUME I. TRANSPORTABLE MAPS SOFTWARE

LIST OF FIGURES

TRANSPORTABLE MAPS SOFTWARE

SECTION ONE

INTRODUCTION AND SUMMARY

Micro-Adaptive Picture Sequencing (MAPS) is a computationally-efficient, contrast-adaptive, variable-resolution spatial image encoding technique. This document describes the implementation of the general MAPS process as a transportable software ensemble which will hereafter be referred to as the TransMAPS Package or simply as 'TransMAPS'.

1.1 Background and Objectives

Micro-Adaptive Picture Sequencing originated in the Information Sciences Division of Control Data Corporation and has undergone extensive further development and exploration with sponsorship from the Rome Air Development Center. These efforts are summarized in the following reports and articles:

References:

RADC MAPS-Related Reports:

LaBonte, A. E. and C. J. McCallum (Control Data Corporation), IMAGE COMPRESSION TECHNIQUES, RADC-TR-77-405, December 1977, Final Technical Report, Contract No. F30602-76-C-0350.

LaBonte, A. E. and T. E. Rosenthal (Control Data Corporation), MAPS IMAGE COMPRESSION, RADC-TR-80-173, May 1980, Final Technical Report, Contract No. F30602-78-C-0253.

LaBonte, A. E. (Control Data Corporation), INFRARED DATA COMPRESSION STUDY, RADC-TR-80-287, August 1980, Final Technical Report, Contract No. F30602-79-C-0080.

SPIE Proceedings Articles:

LaBonte, A. E., "Two-Dimensional Image Coding by Micro-Adaptive Picture Sequencing (MAPS)", Proceedings of the Society of Photo-Optical Instrumentation Engineers, Volume 119, APPLICATIONS OF DIGITAL IMAGE PROCESSING, pp 99-106, 1977.

LaBonte, A. E., "Micro-Adaptive Picture Sequencing (MAPS) in a Display Environment", Proceedings of the Society of Photo-Optical Instrumentation Engineers, Volume 249, ADVANCES IN IMAGE TRANSMISSION II, pp 61-70, 1980.

These prior investigations have dealt with many different facets of the image representation problem including image partitioning, large-area or macro-fidelity control, local or micro-fidelity control, coding strategies, and artifact-masking or perceptual quality enhancement in the reconstructed imagery. Through these studies, MAPS has evolved into a mature set of processes which still retains a rich space of user options. The present effort integrates these previous developments to form the TransMAPS software package.

The general objectives of this implementation are to coordinate existing MAPS results and to make MAPS functionally available to a much wider community of potential users. TransMAPS fulfills these objectives through use of a readily-available language (FORTRAN) running on widely-used computer systems (DEC PDP-11 and VAX). More specific objectives emphasize transportability within this DEC system environment and interactive capabilities to support a broad spectrum of user experience levels and applications intent.

1.2 Software Development

Four main activities were involved in the TransMAPS development cycle. First was evaluation and coordination of the large space of MAPS user options which arose as the technique evolved. Here the principal problems involved assessment of potential conflicts and automation of those features where a clear-cut choice could be made based on the results of prior investigations. Note that overall organization of TransMAPS is determined primarily by the structure which results from this option-space evaluation. Moreover, at least one and sometimes several detailed 'algorithms' are already available for each of the subprocesses invoked by each selected option. Thus, the option assessment really becomes the first stage in the top-down TransMAPS software design.

Again, because of the algorithmic detail already known, the second stage of the development cycle focused on the constraints and compatibility issues raised by the characteristics of the host computer. Here, the sixteen-bit word length of the PDP-11 systems was the dominant controlling limitation. The resulting address field constraints together with the option structure imposed the requirements for extensive partitioning of the MAPS process.

Implementation considerations formed the core of the third major activity group. Issues of modularity, transportable constructs, file environment, and internal program documentation guidelines were principal subtopics. This stage carried development through detailed TransMAPS code preparation.

The fourth phase involved comprehensive program testing followed by TransMAPS installation and demonstration on the target PDP-11/70 in the RADC Image Processing System.

1.3 The TransMAPS Package

The fact that TransMAPS is an integrated software 'package' and not just a collection of computer programs is implicit in all of the development efforts. This package contains seven interrelated main program modules, six data sets, a MAPS standard file structure, a set of pre-planning aids for user interaction support, and the TransMAPS user and maintenance documentation. The contents of these five categories is refined one more level in the following tabulation:

```
TransMAPS Package:
--------- --------

        Seven Main Program Modules:
        ----- ---- ------- --------
                #1  SUBFRM  Raster to Subframe Conversion    -  SF.FSV, SF.OBJ
                #2  MAPS    MAPS Compression                 -  MP.FSV, MP.OBJ
                #3  DMAPS   MAPS Decompression & Level Image Formation
                                                             -  DM.FSV, DM.OBJ
                #4  ADAPT   MAPS Adaptive Image Smoothing    -  AD.FSV, AD.OBJ
                #5  DIFFER  MAPS Difference Image Formation  -  DF.FSV, DF.OBJ
                #6  RASTER  Subframe to Raster Conversion    -  RS.FSV, RS.OBJ
                #7  ANNOTE  Image Assembly and Annotation    -  AI.FSV  AI.OBJ

                Provided on Computer-Compatible Tape (CCT):
                        FORTRAN IV-Plus Source Code  -  x.FSV
                        FORTRAN Object (F4P) Code  -  x.OBJ

        Six Data Sets:
        --- ---- -----
                Annotation Symbol       -Map Tables            -  SYMBOL.BIN
                MAPS Compression Test Image (160 x 128)        -  MTEST.BIN
                Sample MAPS User Parameter Set (Use with MTEST) -  MSET.BIN
                MAPS Product Generation Test Image (120 x 128)  -  GIRL6.BIN
                Two 'Video Frame' Images (480 lines x 624 pixels):
                        Building Scene                          -  BLDGIMG.BIN
                        IEEE Girl                               -  GIRLIMG.BIN

                Provided on Computer-Compatible Tape (CCT)

        MAPS File Structure with Standard Filenames and Headers:
        ---- ---- --------- ---- -------- --------- --- -------
                Source Image (One Subframe/FIXED Record)     -  IMAGE.DAT
                User Parameter Set                           -  MSET.DAT
                MAPS Compression Stream (FIXED Records)      -  MAPS.DAT
                MAPS Block/Pattern Image (Subframes)         -  DMAPS.DAT
                MAPS Resolution Image (Subframes)            -  LEVEL.DAT
                MAPS Adaptively Smoothed Image (Subframes)   -  ADAPT.DAT
                MAPS Difference Image (Subframes)            -  ERROR.DAT
                Fidelity Performance Summary (Listing)       -  EPRINT.DAT
                MAPS Product Image (Raster)                  -  xRAST.DAT
                                                                x = I,D,L,A,E
                Annotation Symbol-Map Tables                 -  SYMBOL.DAT
                Annotated Image (Raster)                     -  ANIMG.DAT
                Annotated Printer Pseudo-Image (Listing)     -  APRINT.DAT

        User Interaction Pre-Planning Aids:
        ---- ----------- ------------ -----
                MAPS Planning Form  -  Compression
                MAPS Planning Form  -  Product Generation
                Annotation and Image Assembly Planning Form

        Documentation:
        --------------
                TransMAPS User's Manual
                TransMAPS Maintenance Manual
```

The first six program modules all implement processes directly related to MAPS. This set of six further divides into modules #1 and #2 which deal with compression and modules #3 through #6 which provide the generation of MAPS output 'products'. Four product types are available: MAPS directly-decompressed tonal images, MAPS resolution or level images, MAPS adaptively-smoothed tonal images, and MAPS difference images. The 'level' images give direct display of the pattern of variable resolution generated by the MAPS compression process. The 'difference' images display the fidelity between source and product images in terms of either a 'signed' error with a neutral gray zero-point bias or an 'amplified' absolute error.

The seventh module provides a stand-alone image assembly and annotation capability. It is included to support the display of the MAPS products but can be used to format and label imagery from other sources as well. Figure 1-1 exhibits a source image, its MAPS tonal decompression, the corresponding resolution image, and an amplified difference image assembled and annotated using module #7. The original image in this example is 480 lines x 624 pixels x 8 bits; the MAPS compression level is 2 bits/original pixel; and the difference amplification factor is 10. In the resolution image, successively lighter regions correspond to finer levels of local MAPS resolution. The difference image has been complemented (during the assembly process) so the lighter flecks represent larger absolute source-to-MAPS differences. The sample in Figure 1-1 is intended to give the flavor of the capabilities of the TransMAPS package and to illustrate them with 'real-world' imagery (in this case a frame of 'video' size).

1.4 Documentation Organization

The complete documentation for TransMAPS is distributed across three volumes: this final report, the TransMAPS User's Manual, and the TransMAPS Maintenance Manual. The User's Manual emphasizes the basis of

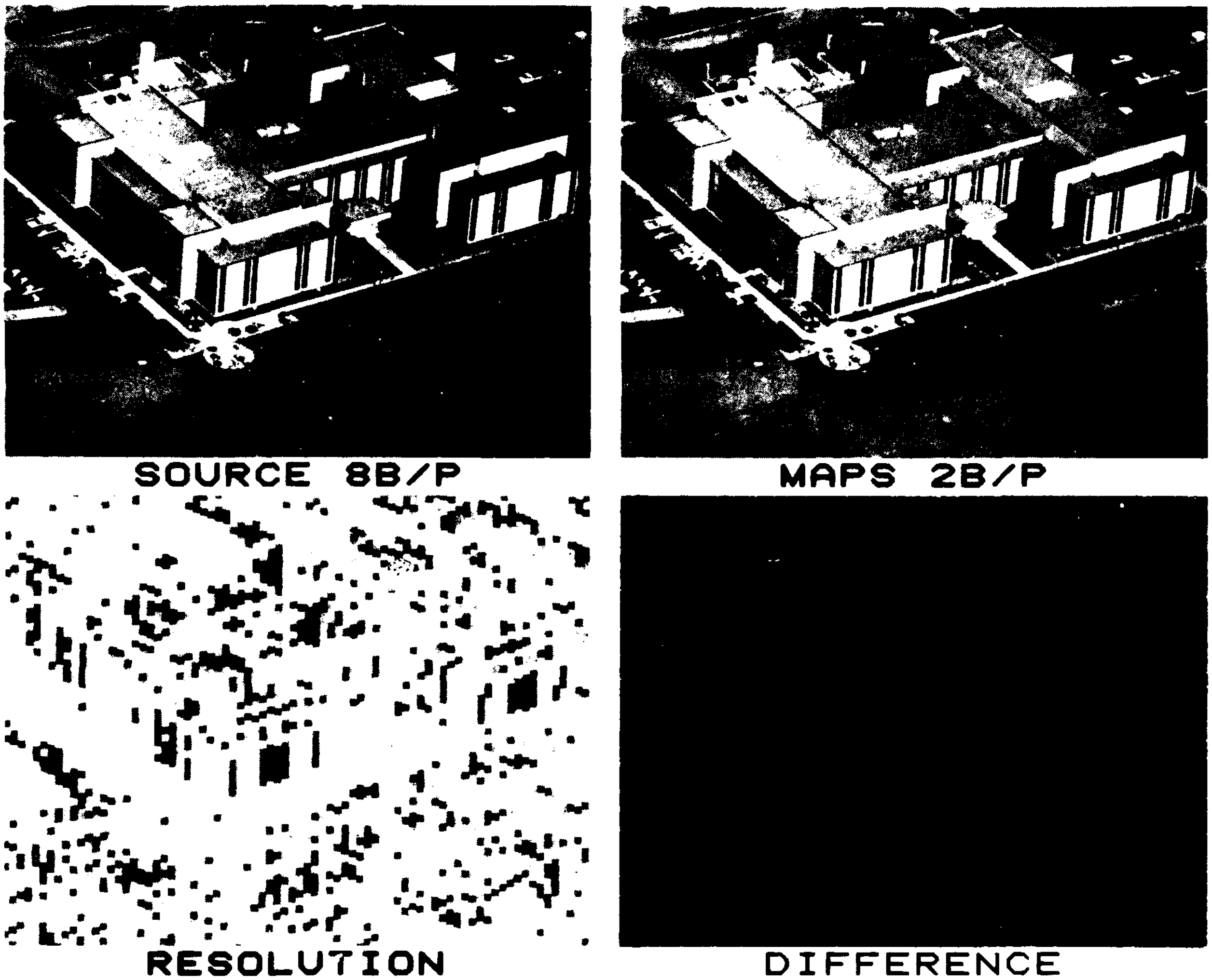


Figure 1-1. Example of TransMAPS Software Application

the MAPS subprocesses, the space of user options, and the interactive protocols needed to apply the package as an image coding system. The focus in the User's Manual is thus on MAPS as an integrated family of techniques and on the concepts which underlie them. The user may either apply MAPS for functional image compression or may use TransMAPS to become familiar with and explore the MAPS conceptual and process structure. The perspective here is on 'what it does'.

The Maintenance Manual emphasizes the system and specific implementation aspects of TransMAPS. It addresses MAPS as hosted by the computer system and focuses on installation, modification, and code maintenance issues. The Maintenance Manual contains complete COMMENT-annotated source listings of the seven program modules and these listings form the ultimate level of detailed implementation documentation. The perspective here is on 'how it's done'.

The present document - the project final report - is presumed to contain both the User's Manual and the Maintenance Manual in its scope. They are simply published as separate documents for the convenience of later use where their roles are quite different. The remaining topics which must be covered in this document, then, involve descriptions of the activities specific to the original construction of TransMAPS; in essence, the 'scaffolding' of the TransMAPS development. Four topics - user option assessment, process partition, implementation, and testing - are discussed briefly in the sections which complete this volume. Readers unfamiliar with the basic MAPS constructs may wish to read the synopsis of these processes in Section Six of the User's Manual to gain a framework for the rest of the documents. Alternatively, the references outlined in Section 1.1 may be consulted for this purpose.

SECTION TWO

USER OPTION ASSESSMENT

The organization of TransMAPS is determined primarily by the structure imposed on the space of user options. Throughout previous development and refinements of the MAPS technique, many alternate configurations and strategies were investigated. This resulted in a very large space of potential user options. Thus, the first task in the TransMAPS integration involved three subtasks focused on this option space: reduction of the space by 'automating' some of the selections; sequencing and 'grouping' the retained options to structure the process; and assessing the 'compatibility constraints' within this structure. Each of these subtasks will be dealt with in turn.

2.1 Automated Selections

The variety and nuances of possible MAPS alternatives is bewildering. Thus, any reduction in this option space which does not significantly degrade flexibility or performance will enhance the usability of TransMAPS. Two types of option selection 'automation' were sought. In the first type, the selection is made implicit as a part of another user action so that it becomes transparent. In the second type, the range of states for a particular option dimension is restricted to those where clear advantages have been demonstrated.

Three subclasses of implicit selections arose - in image staging, in process mode, and in generic options. In image staging, the DEC systems are heavily disk-file oriented and do not support simple direct magnetic tape interaction under FORTRAN. Thus, file skipping and positioning occur only on transfer of image files from tape to disk and not on every processing pass through a given source file. Moreover, communication of

image data between processes is disk-based and this communication can be made transparent by implementation of a MAPS file structure with standard naming conventions. The staging tasks are then handled either automatically or as part of the initial data entry and need not be made explicit.

Specification of the MAPS process mode involves selection of the process(es) to be run on the source imagery and their sequence. These choices are made implicitly as the user invokes the corresponding TransMAPS program modules. Furthermore, since intermodule communication is accomplished through the standard file structure, the user gains flexibility in sequence specification because the required intermediate results are generated and saved automatically. Thus, the full mode selection need not be developed beforehand - invoking TransMAPS modules remains interactive.

Generic options allow more specific option types to be absorbed as special cases and thus simplify the explicit tree of options without reducing flexibility. One potential option absorbed in this manner is the 2x2-element-sharpening heuristic from the adaptive smoothing module which is superseded by the more powerful subelement pattern coding which applies to 2x2 and larger MAPS elements if desired. Even more generality is represented by the use of piecewise-linear code-space-to-constrast-space and code-space-to-intensity-space remapping specifications. More specific options such as log or exponential remappings can be approximated by and thus absorbed into the process of defining suitable breakpoints for the piecewise-linear mappings. Specification of the required intensity-space-to-code-space demapping is automated by inverting the code-to-intensity remap.

Five subclasses of state-restriction selections arose - in file annotation, in performance evaluation, in resolution image generation, in pattern bias specification, and in adaptive smoothing control. Since the

MAPS file structure must serve intermodule communication, certain ancillary information such as image size and partition parameters must be included as part of the file information. Thus, some form of file header is required and a standard header format was chosen. This format includes an 'image name' field and is added automatically to all of the standard MAPS files. Thus, the user no longer need choose whether to provide such annotation or not.

Performance evaluation for both compression level and fidelity in overall image terms is a simple by-product of the statistics which must be gathered for determining optimal pattern biases. Thus, these evaluations have a negligible impact and are automated as part of the compression process. Gathering data for more detailed fidelity distributions, however, does have a significant impact on compression computational efficiency. Moreover, if adaptive smoothing is to be applied, the process is sufficiently non-linear so that difference statistics can only be gathered during or after the smoothing process. Thus, more detailed fidelity performance analysis is deferred until the point where MAPS difference images are formed. These statistics are easily obtained there so they are included as an automatic rather than optional by-product in the difference process.

The MAPS resolution or level image is a useful evaluation tool to assess how MAPS is distributing its resources relative to the image content. The information in this resolution image is also a necessary input to the MAPS adaptive smoothing process. Thus, formation of the level image is conveniently automated as part of the direct MAPS tonal image decompression. The only remaining potential option in this case, then, is the selection of the gray-scale values to be associated with each level. This association must present the 'level' image itself in a discernible form and must make the information easily accessible for

adaptive smoothing. A suitable selection which automates the gray-scale assignment and meets these requirements is one in which the MAPS level or resolution codes are simply stored in the upper three bits of each level image byte.

Selection of appropriate pattern biases to be used with the MAPS subelement pattern coding mode can be optimized in a mean-square-error sense and automated based on statistics gathered during the compression step. Thus, the imagery itself yields the best choice for these bias values and the user need not be burdened with their separate specification.

In MAPS adaptive smoothing, the 'adaptation' includes three different considerations: the size of the 'convolution' window, activation of surrounding elements based on their size relative to the target element, and activation of surrounding elements based on their intensity relative to the target element. Although specification of each of these could be left to the user's control, specific choices perform well and have significant a priori justification. Thus, the window size is chosen to have an edge which is one cell smaller than the target element. This makes the window 'odd' (symmetric about its center pixel). This choice is consistent with the assertion that the target element size reflects the local intensity correlation length in the image. The 'surround' activation based on size is chosen to include all elements no smaller than one resolution level below the target element. This is consistent with the assertion that finer relative resolution is a priori evidence of localized image activity and should not be included in the 'convolution'. Finally, the selection of weighting functions for the convolution is restricted to a choice between uniform weight over the window or two-dimensional Gaussian weight with user specified spread. Uniform weighting has a particularly simple implementation and Gaussian weighting exhibited slight superiority among the various functions investigated in previous MAPS studies.

2.2 Option Groupings

Following the pruning of the option tree, the next major task in the user option assessment is to order the remaining options and to cluster them into groups according to the processes which require them. The ordering chosen is essentially the sequence in which the option selections are used. The process groupings are as follows:

```
Option Groupings:
------ ---------

        Raster to Subframe Conversion:
        ------ -- -------- ----------
                Source Image Identification
                Source Image Position Specification
                Source Image Size Specification
                Source Image Partition Specification

        MAPS Compression:
        ---- -----------
                User Mode Selection
                Macro-Fidelity Control
                Micro-Fidelity Control
                Gray-Scale Manipulations

        MAPS Decompression and Resolution Image Formation:
        ---- ------------- --- ---------- ----- ---------
                (User Transparent)

        MAPS Adaptive Image Smoothing:
        ---- -------- ----- ---------
                Convolution Weighting Specification
                Dither Amplitude Specification

        MAPS Difference Image Formation:
        ---- ---------- ----- ---------
                Input Image Pair Selection
                Difference Image Control

        Subframe to Raster Conversion:
        -------- -- ------ ----------
                Output Product Image Type Selection

        Image Assembly and Annotation:
        ----- -------- --- ----------
                Output Image Specification
                Embedded Input Image Specifications
                Embedded Annotation Specifications
```

Note that this tabulation displays only the first level of refinement under each process. Particularly in the MAPS Compression and the Image Assembly and Annotation processes, several further levels are needed.

This option hierarchy coupled with an appropriate default structure allows the casual or inexperienced user to treat TransMAPS as a 'black box' (or more nearly dark gray) image coding system. Very few parameters need to be specified for simple uniform fidelity coding and reconstruction of the source imagery. On the other hand, the more experienced user or one who seeks to gain more insight into the MAPS processes can selectively penetrate the option tree and tailor the control as desired.

In order to assist with such exploitation or exploration, the option space has been laid out in three detailed 'User Planning Forms'. They are intended to serve as an aid for pre-planning of complex interactive sessions, to provide a convenient record of MAPS processing, and to form a 'road-map' which lays out the entire option space at one time. Reduced versions of these Planning Forms are exhibited here as Figures 2-1 (MAPS Compression), 2-2 (MAPS Product Generation), and 2-3 (Annotation and Image Assembly).

The information on these three forms, then, characterizes the option structure and thus the organization of the interactive TransMAPS package. Further descriptions of the individual option entries is given in the User's Manual, Section Seven.

2.3 Compatibility Constraints

The remaining option analysis effort was directed at uncovering combinations of active options which are internally incompatible. Only two significant problems were found and neither restricts the performance of the package appreciably. Both problems involve the 'staggered' subframe image partition. Both also arise partially due to 'computational considerations' so they are not strict option incompatibilities.

MAPS PLANNING FORM (COMPRESSION) (ALLOWED RANGE) [DEFAULT] DATE ____________

MODULE **USER INPUTS**

RUN SUBFRM

FILES:

IN - USER RASTER

OUT - IMAGE.DAT

SOURCE IMAGE FILE NAME (_ _ _ _ _ _ _ _ _ _ _)

USER IMAGE NAME (_ _ _ _ _ _ _ _ _)

SKIP LINES [0]________ SKIP PIXELS [0] ________

PROCESS LINES[ ]________ PROCESS PIXELS[ ] ________

BITS/PIXEL (6 8) [8]

SUBFRAME SIZE (8 16 32) [8] SUBFRAME GRID (SQUARE STAGGER) [SQUARE]
8X8 ONLY

RUN MAPS

FILES:

IN - IMAGE.DAT

OUT - MAPS.DAT

IN/OUT - MSET.DAT
USER PARAMETERS

MODE (QUICK USER FULL) [Q]

MACRO FIDELITY CONTROL:

IMAGE PARTITION:

PIXEL INDEX DIRECTION →

LINE INDEX DIRECTION ↓

EACH CELL (1 2 3 4) [ALL 1]

MICRO FIDELITY CONTROL:

PARAMETERS	GROUP 1	GROUP 2	GROUP 3	GROUP 4
CONTRAST SCALE	____	____	____	____
TAPER	[3.0]____	[3.0]____	[3.0]____	[3.0]____
STEP FRACTION	[0.5]____	[0.5]____	[0.5]____	[0.5]____
STEP BIAS	[0.1]____	[0.1]____	[0.1]____	[0.1]____

OR:

MATRIX	E	M	L	U	E	M	L	U	E	M	L	U	E	M	L	U
0-1																
1-2																
2-3																
3-4																
4-5																

BLOCK/PATTERN ASSIGNMENT, EACH LEVEL (B P) [BPPBBB]
012345 LEVEL

GRAY-SCALE MANIPULATIONS:

CONTRAST REMAP: CODE CONTRAST

PIECEWISE LINEAR BREAKPOINT PAIRS (NON-DECREASING)

CODE (0-255)

CONTRAST (0-4095)

CODE	CONTRAST
0	0
[255]____	[255]____
____	____
____	____
____	____
____	____
____	____
255	____

INTENSITY REMAP: CODE INTENSITY

CODE (0-255)

INTENSITY (0-4095)

CODE	INTENSITY
0	0
[255]____	[255]____
____	____
____	____
____	____
____	____
____	____
255	____

INTENSITY RESET:

(MEAN)
(PSEUDO-MEDIAN)
(LOWEST)
(SECOND)
(THIRD)
(HIGHEST)

[MEAN]

RESULTS: COMPRESSION: ____________

MEAN SQUARE ERROR: ________%

LEVEL:	0	1	2	3	4	5
MAPSELS:	____	____	____	____	____	____
OPTIMAL BIASES: −	____	____	____	____	____	____
+	____	____	____	____	____	____

Figure 2-1. MAPS Compression Planning Form

MAPS PLANNING FORM (ALLOWED RANGE) [DEFAULT]
(PRODUCT GENERATION)

DATE________________
IMAGE NAME________________
COMPRESSION________________

MODULE		USER INPUTS
RUN DMAPS		
FILES:		
IN - MAPS.DAT		
OUT - DMAPS.DAT		
OUT - LEVEL.DAT		
RUN ADAPT /NOT WITH STAGGER GRID/		
FILES:		CONVOLUTION WEIGHTING (GAUSSIAN UNIFORM) [G]
IN - DMAPS.DAT		SIGMA MULTIPLE AT WINDOW CORNER [2.0] ____ /GAUSSIAN ONLY/
IN - LEVEL.DAT		RANDOM DITHER AMPLITUDE [4.0] ____
OUT - ADAPT.DAT		
RUN DIFFER		
FILES:		FILE PAIRINGS (IMAGE vs DMAPS IMAGE vs ADAPT DMAPS vs ADAPT) [I D]
IN - IMAGE.DAT	ANY TWO	DIFFERENCE PARAMETER: (<0 SIGNED =0 STATISTICS ONLY >0 AMPLIFICATION FACTOR) [10]____
IN - DMAPS.DAT	ANY TWO	
IN - ADAPT.DAT	ANY TWO	
OUT - ERROR.DAT		
OUT - EPRINT.DAT (LISTING)		
RUN RASTER		
FILES:		MAPS RASTER PRODUCT (IMAGE DMAPS LEVEL ADAPT ERROR) [DMAPS]
IN - IMAGE.DAT	ANY ONE	
IN - DMAPS.DAT	ANY ONE	
IN - LEVEL.DAT	ANY ONE	
IN - ADAPT.DAT	ANY ONE	
IN - ERROR.DAT	ANY ONE	
OUT - IRAST.DAT	TYPE MATCHES INPUT	
OUT - DRAST.DAT	TYPE MATCHES INPUT	
OUT - LRAST.DAT	TYPE MATCHES INPUT	
OUT - ARAST.DAT	TYPE MATCHES INPUT	
OUT - ERAST.DAT	TYPE MATCHES INPUT	

Figure 2-2. MAPS Product Planning Form

ANNOTATION AND IMAGE ASSEMBLY PLANNING FORM

RUN ANNOTE

DATE ____________

OUTPUT IMAGE:

TYPE (RASTER FILE PRINTER LISTING) [R] LINES ________ PIXELS ________ COMPLEMENT (Y N) [N]
ANIMG.DAT APRINT.DAT

EMBEDDED IMAGES: NUMBER (0 1 2) [0]

IMAGE	FILENAME	INPUT POSITION SKIP LINES	INPUT POSITION SKIP PIXELS	SIZE LINES	SIZE PIXELS	OUTPUT POSITION START LINE	OUTPUT POSITION START PIXEL	COMPLEMENT
1	----------	[0]____	[0]____	____	____	____	____	(Y N)
2	----------	[0]____	[0]____	____	____	____	____	(Y N)

EMBEDDED MESSAGES: NUMBER (0 - 20) [0]

MSG	ORIENT	CHAR	SYMBOL SIZE	CENTER LINE	CENTER PIXEL	COMP	TEXT 1 5 10 15 20 25 30 35 40 45 50
1	(TBLR)	___	(1234)	____	____	(Y N)	--
2	(TBLR)	___	(1234)	____	____	(Y N)	--
3	(TBLR)	___	(1234)	____	____	(Y N)	--
4	(TBLR)	___	(1234)	____	____	(Y N)	--
5	(TBLR)	___	(1234)	____	____	(Y N)	--
6	(TBLR)	___	(1234)	____	____	(Y N)	--
7	(TBLR)	___	(1234)	____	____	(Y N)	--
8	(TBLR)	___	(1234)	____	____	(Y N)	--
9	(TBLR)	___	(1234)	____	____	(Y N)	--
10	(TBLR)	___	(1234)	____	____	(Y N)	--
11	(TBLR)	___	(1234)	____	____	(Y N)	--
12	(TBLR)	___	(1234)	____	____	(Y N)	--
13	(TBLR)	___	(1234)	____	____	(Y N)	--
14	(TBLR)	___	(1234)	____	____	(Y N)	--
15	(TBLR)	___	(1234)	____	____	(Y N)	--
16	(TBLR)	___	(1234)	____	____	(Y N)	--
17	(TBLR)	___	(1234)	____	____	(Y N)	--
18	(TBLR)	___	(1234)	____	____	(Y N)	--
19	(TBLR)	___	(1234)	____	____	(Y N)	--
20	(TBLR)	___	(1234)	____	____	(Y N)	--

Figure 2-3. MAPS Image Assembly and Annotation Planning Form

The first difficulty is among the staggered subframe partition, the larger subframe sizes (16x16 and 32x32), and the maximum array size addressable on a sixteen-bit computer. The essence of the problem is that a 16 or 32 line recirculating buffer is too restrictive on the source image size, and implementation in terms of an 8-line recirculating buffer with successive partial subframe extraction is too disk I/O intensive to achieve a reasonable efficiency level. However, the advantages of a staggered subframe grid in reducing the perceptible 'blockiness' in the reconstructed image are mostly lost in the larger subframes anyway since 16x16 and 32x32 block sizes can usually be discerned independent of the characteristics of their surroundings. Thus, restriction to 8x8 subframes for the staggered grid partition is not a serious operational limitation.

The other difficulty is among the staggered subframe partition, the adaptive smoothing process, and the 'fast' algorithm for the required 'convolution'. This fast algorithm gains a major portion of its efficiency from the fact that the active elements surrounding the target element have invariant sampling point positions given the size (resolution level) of the target. In the staggered mode, the 'surround' for elements along the subframe edges is different from the surround for an element interior to the subframe. Thus, this crucial invariance is lost and a much less efficient sampling strategy would be required. Thus, adaptive smoothing is restricted to square grid subframe partitions. Note, however, that subframe stagger and adaptive smoothing are really competing processes which both seek to decrease the perception of 'blockiness' in the reconstructed MAPS imagery. Stagger has very low computational cost but is only partially effective in hiding 'blockiness'. Adaptive smoothing is very effective in perceptual improvement but also incurs relatively higher computational expense. Adaptive smoothing combined with stagger seems unlikely to give significantly better perceptual quality than is achieved by adaptive smoothing on square grids. Again, no appreciable operational capability appears to be lost through this constraint.

SECTION THREE

PROCESS PARTITION

The sixteen-bit word size of the PDP-11 systems introduces three types of constraint on the TransMAPS development. The three types are: task space limitations, array address limitations, and system overhead costs. Together, these constraints dictate the need to partition MAPS into process modules which can be accommodated individually within the PDP-11 operating environment. The next three subsections discuss the implications of the limitations; the final subsection then describes the partition and the associated intermodule communication structure.

3.1 Task Space Limitations

Even though a typical large PDP-11 system may have several hundred thousand bytes of main memory, individual tasks are limited to 32K if standard FORTRAN constructs are employed. In 'mapped' PDP-11 systems, this restriction can be circumvented by the process of 'windowing' in 4K blocks (with some overhead for window maintenance). In general, such windowing is not directly accessible via FORTRAN except for data space extension through the VIRTUAL array declaration. Although VIRTUAL allows data access outside of the normal 32K task space, it does so at some expense in program efficiency. Thus, the VIRTUAL construct should be exploited only where absolutely needed.

Another strategy for handling programs which exceed the 32K task limit is through the use of overlay structures. However, this approach is appropriate where the excess size is due to executable code which can be overwritten when its functions have been completed. In the TransMAPS process (as with many 'image processing' tasks), the size requirements are dictated more by the image data needs than by the program statements to manipulate that data. Furthermore, many of the subprocesses are

table-driven with table space dominating the size of the task. Again, this is more data-like than program-like so overlays are of marginal utility.

Four guidelines emerged from these considerations:

- Restrict code to FORTRAN constructs, if possible, to simplify transportability and program maintenance;
- Attempt explicit process partition into modules which reflect the logical structure of MAPS and avoid the complexity inherent in overlays;
- Minimize use of the non-standard and relatively slow VIRTUAL declaration wherever possible; and
- Exploit process symmetries to reduce table size for table-driven processes.

3.2 Array Address Limitations

The sixteen-bit word size also limits the maximum number of addressable elements in an array to 32K. Elements could be single-byte, two-byte, four-byte, or eight-byte. However, the basic source image pixel size is single-byte (8 bit or 6 bits right-justified in 8). Thus, image manipulations are much easier to understand if the arrays are directly organized into bytes. But this leads to a problem if 32x32 pixel subframes are to be used since 32 lines of data must be accessed to obtain each subframe. Straightforward implementation of a 32-line buffer would then limit the image edge (line length) to a thousand pixels. This is clearly too restrictive for a general MAPS capability.

Fortunately, both PDP-11 file systems - FCS-11 and RMS-11 - support a DIRECT access capability for binary files of FIXED record length. This makes it possible to develop input and output modules which reorganize the raster source imagery to subframe format by extracting segments and updating subframes from successive swathes of lines. An eight-line swath at four thousand pixels per line will still fit within the 32K array address constraint. This sets an acceptable limit (4000 pixels) on the size of the source image.

Once in subframe format, all subsequent TransMAPS modules can then deal with the data in terms of deterministic record (subframe) size. These sizes are 64 bytes (8x8 subframe), 256 bytes (16x16 subframe), and 1024 bytes (32x32 subframe). Even the largest of these is a small fraction of the available 32K task space. Moreover, all three sizes are sub- or supra-multiples of the basic disk sector length of 512 bytes. The DIRECT mode of disk access exploits this in finding the track and sector for a particular subframe index; a significant efficiency advantage.

3.3 System Overhead

The 32K task space must also accommodate some system overhead in the form of data communication buffers. The space set aside for this purpose is determined by two Task Build parameters - ACTFIL and MAXBUF. ACTFIL determines the maximum number of files which can be active (open) at one time. MAXBUF determines the maximum record size which can be handled on any file. The buffer space is determined by the product of these two. Thus, it is important to restrict the number of files to the minimum needed and to open, read, and close any initial table-loading files before opening image handling files in each module.

Furthermore, where possible, it is desirable to employ file types with a minimum physical RECORDSIZE. For modules employing subframe data in the DIRECT access mode, this size is at least 1024 bytes. For raster organized image files, the record size will depend on the RECORDTYPE.

Another Task Build parameter, UNITS, also affects the system overhead charged against the task space. UNITS must be set at least as large as the largest logical unit number employed in the program. File table space is set aside for each possible logical unit number up to UNITS whether each number is used or not. Thus, use of the small numbers, 1 through 4, for the data files in each module is advantageous. Unit 5 is the default terminal designation and Unit 6 is the default printer value; retention of these seems reasonable for standardization. Extension

beyond the default, UNITS=6, does not appear necessary.

3.4 Intermodule File Communication

The logical partition of the MAPS processes is already supported by the user option evaluation results. MAPS compression, MAPS product formation, and Image Assembly and Annotation form the three major process categories. Compression is further divided into the macro step of subframe reorganization and the micro step of MAPS coding within each subframe. Product formation is also further divided into MAPS decompression with resolution image formation; adaptive smoothing; difference image formation; and reformatting back to raster organization. In summary, a suitable logical partition into seven modules (including process mnemonics) is:

#1	SUBFRM	Source raster to subframe conversion;
#2	MAPS	MAPS subframe compression;
#3	DMAPS	MAPS decompression/level image formation;
#4	ADAPT	MAPS adaptive smoothing;
#5	DIFFER	MAPS difference image formation;
#6	RASTER	Subframe to raster conversion; and
#7	ANNOTE	Annotation and image assembly.

The user interaction with each of these modules was outlined in Sectio. Two. However, appropriate image data must also interact with each of these modules and this communication has several requirements. The process should be transparent to the user. This implies the need for standard (dedicated) file names which can be opened automatically when a particular process is invoked. It also means that necessary control parameters from previous processes be carried internal to the file; a standard header format meets this latter need. Moreover, the file names should be mnemonic for the file type but sufficiently unique to avoid confusion with other files likely to reside in the system. Such a combination of dedicated mnemonic file names enables effective and efficient file maintenance activities to be carried out on the TransMAPS data environment even if they are generated over an extended period of time.

A standard configuration of twelve file types was developed for this intermodule communication function. They are exhibited along with the corresponding modules in Figure 3-1. This figure presents the complete macro-structure of TransMAPS; it is the key organizing chart for the entire software package.

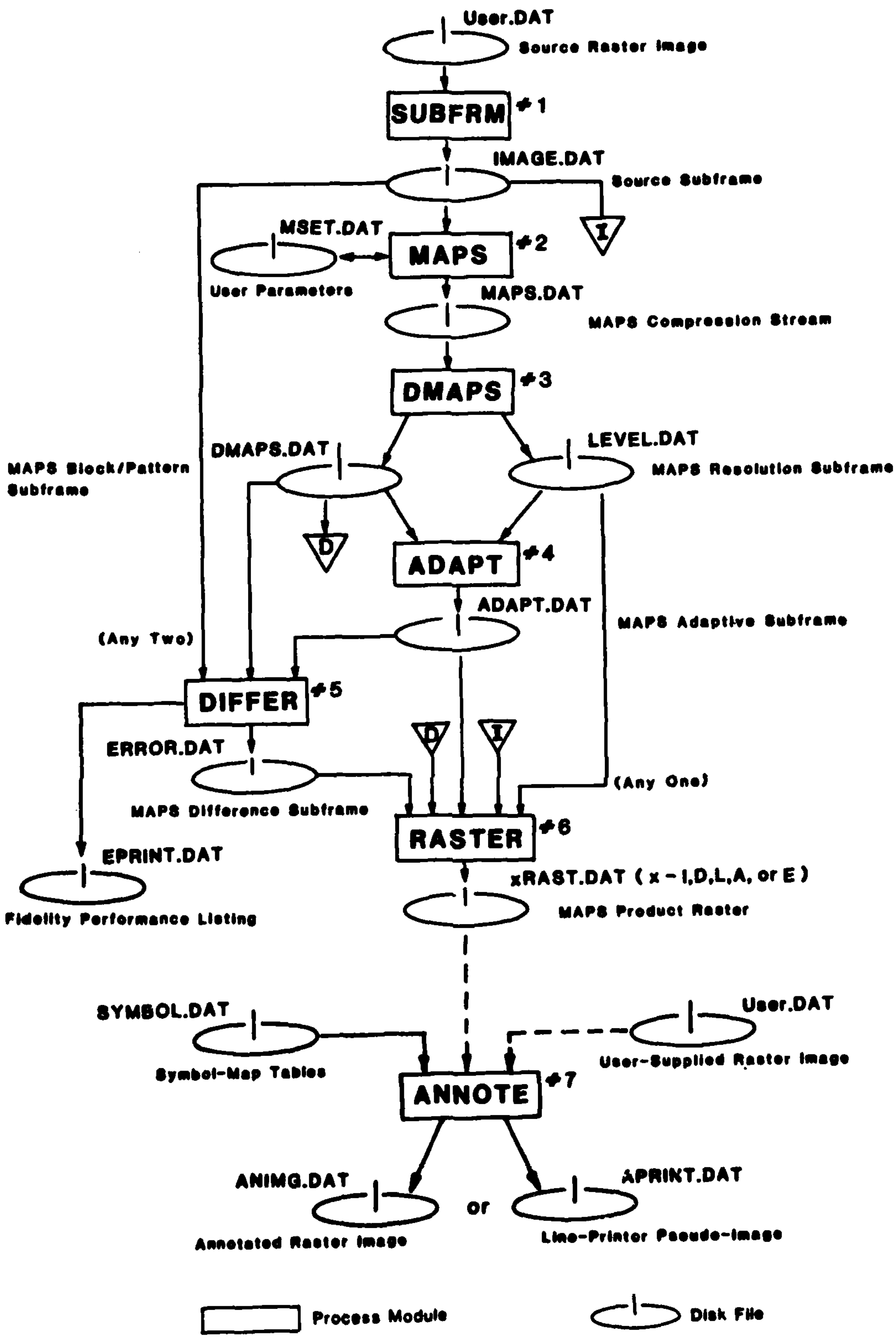


Figure 3-1. Overall TransMAPS Structure

SECTION FOUR

IMPLEMENTATION

Detailed program module development is the next broad step in the top-down design which implements the structure of TransMAPS given in Figure 3-1. Here, 'working models' for many of the process steps already existed in the various MAPS software elements available from prior internal Control Data implementations. Thus, much of the effort consisted of coordination and integration, and the TransMAPS implementation phase could emphasize the issues of transportability and its close relative, software maintainability. Both maintenance and transport are aided by use of standard constructs, modularization of code, standardization of file structures, and consistent internal program documentation practices. Each of these areas is discussed briefly in the remainder of this section.

4.1 Transportability

Some of the transportability considerations have already been mentioned; an example is the use of file types supported by both FCS-11 and RMS-11 file systems. More generally, coding was restricted to FORTRAN constructs supported in both DEC compilers - FORTRAN and FORTRAN IV-PLUS - insofar as possible. This restriction was imposed even where small penalties in execution efficiency were known to accrue.

In two instances, TransMAPS employs constructs which only FORTRAN IV-PLUS supports. In ANNOTE, the image assembly and annotation module, extensive use is made of the library shift function, IISHFT. The shift allows very rapid manipulation of the annotation character bit maps in the various resampling operations; any other approach seemed unnecessarily obscure and cumbersome.

The other type of construct limited to FORTRAN IV-PLUS involves use of

Integer*4 (i.e. four byte integer) arithmetic in several calculations needed for performance evaluation and element or subframe counts. Simple FORTRAN supports the Integer*4 data type but only allows Integer*2 arithmetic operations. Although the four-byte arithmetic can be partitioned into two-byte arithmetic with extensive overflow checking, the efficiency penalties seem far in excess of the slight gains in transportability. In any event, such two-byte for four-byte replacement can be treated as a maintenance function in the rare instances where it is required by the system.

4.2 Modularization

To support maintenance activities and possible future software modification or extension, the TransMAPS detailed design is extensively modularized. Including the main routines, the seven principal modules contain a total of sixty-eight routines. The average routine length without COMMENTS but including continuations is less than fifty lines. Although most routines fit on a single page (at least before COMMENTing), the code for the user interactions was not so restricted. For each module, the interactive protocols were grouped in one (or two) routines for ease in location rather than being broken up arbitrarily to achieve artificially short code segments. Moreover, the FORMAT statements are placed at the point of use rather than being collected at the beginning or end of the routine. This allows the interaction to be followed sequentially through its steps by anyone reading the code and should contribute significantly to rapid understanding of its flow.

Each interaction is in the form of a 'TYPE' statement (with FORMAT) which queries the user and gives current default values and allowed ranges where appropriate. This prompt is followed immediately by the corresponding 'ACCEPT' statement which processes the user's response. If applicable, validity checking is included following the response; such data checks are typically set off by indenting the corresponding code in the user interaction routines. Thus, the interaction proceeds in

consistent chunks with the FORMATs providing integrated internal documentation. Except to set off major groups of user parameters, further COMMENTs in these routines were thought to be more intrusive than helpful and were not used.

All routines were restricted to a single entry and a single exit point. Within the routines, flow is either simple sequential, simple conditional (the equivalent of 'if-then-else'), simple iteration ('DO-loop'), or the non-linear recursion implied by the MAPS resolution adaptation. Thus, except for the MAPS recursive sequencing, all control constructs are of the classical 'structured' variety. The MAPS sequencing is also simple and clear from context.

In many instances, iterative control involves several levels of nesting. For these cases, the hierarchy is set off by extensive and consistent use of DO-loop indentation in all TransMAPS routines other than those for user interaction as already described.

4.3 Raster Image File Formats

As noted earlier, DEC FORTRAN does not support direct magnetic tape operations in any simple manner. Thus, the communication of image data to and from the TransMAPS package is via disk file. Since there does not appear to be any generally-accepted standard format for raster image files, a specific file structure was chosen to satisfy other conditions in TransMAPS. In particular, TransMAPS expects input raster images to be in the form of sequential binary files with a SEGMENTED record type. The advantage of this form is that large logical records (up to four thousand bytes) can be accommodated without the necessity to handle correspondingly large physical records. Thus, the buffer space does not have to be allocated on the basis of this largest logical record size and MAXBUF need not be set larger than 1024, the largest expected record of FIXED type. This limit on the buffer overhead, in turn, has allowed TransMAPS to be implemented without resorting to the VIRTUAL declaration

in any of the modules; a significant gain in efficiency! Even with this restriction, three of the seven modules are well over 31K in required task space. Hence, the restriction on raster file format is a necessary one to gain this benefit.

4.4 Subframe Image File Formats

Sequential binary files of FIXED record type were selected as the format for the subframe-organized images (and for the stream of MAPS-compressed data). This choice allows use of the DIRECT access mode which in effect gives random image access at the individual record level. Several benefits accrue from this. In the process of conversion from raster to subframe organization, groups of raster lines can be partitioned into partial subframe segments and the corresponding subframe records retrieved and updated individually as needed. An analogous process in the conversion from subframes back to raster lines is also supported by this random access to individual subframe records.

In the adaptive smoothing process, it is necessary to have access to the data from subframes bordering the subframe of current target interest. Again, a random access capability which allows retrieval of individual subframes is required; DIRECT access mode also supports this need.

Finally, some data in the file header is not available until all following records have been completed. The DIRECT access mode allows this first record to be inserted in the file at the end of the process and does so in a simple fashion.

As noted earlier, both DEC file systems - FCS-11 and RMS-11 - support such FIXED record type DIRECT access binary files. Thus, TransMAPS transportability is not compromised.

4.5 COMMENT Guidelines

For a competent FORTRAN programmer, the most useful internal program documentation is the flow of the executable code itself. Once program intent and basic definitions are understood, COMMENTing within the flow can often be more distracting than helpful. Thus, a consistent philosophy of program COMMENTs for TransMAPS has been employed which seeks to aid without intrusion.

File communication, overall intent and structure, and definitions of key variables are grouped at the beginning of each of the seven main modules. Each subroutine, then, is provided with a much briefer heading which also narrates intent and describes local CALLing links. In-line COMMENTs are restricted to setting off major blocks for quick location. On-line flags (using the DEC separator convention '!') are used to locate points where default values are set and to note points at which I*4 arithmetic is carried out. These flags are supplied to simplify site-specific maintenance and installation changes.

The modularization within TransMAPS is dictated more by the need for logical refinement into 'graspable' chunks than by requirements for code segments which are used repeatedly at different points in the flow. Thus, intermodule communication need not have extensive lists of formal parameters which take on different values at different call points. Rather, the various routines tend to work on a common body of data and use a common set of control parameters. Thus, the communication problem is handled by extensive use of named COMMON blocks. Each main module contains a complete description of the COMMON blocks with variable names, data types, and role characterization. In effect, this description becomes a data dictionary for the module.

The COMMENT formats are summarized schematically in the following tabulation:

```
COMMENT Formats:
------- --------

    Principal Module Headers:
    --------- ------ --------
        C
        C  +----------------------------------------+
        C  |                                        |
        C  |  TransMAPS Module #n:  process descriptor  |
        C  |                                        |
        C  +----------------------------------------+
        C
        C                        Control Data Corporation - 1982
        C
        C Files: Unit  Name  Content            From/To  Type
        C ------
        C  In/Out  n   zzzzz descriptor         module   SEGMENTED
        C                                            or DIRECT
        C                                            or FIXED SEQ.
        C                                            or FORMATTED
        C
        C User Interaction:
        C -----------------
        C       principal interactive parameter groups
        C
        C Program Structure:
        C ------------------
        C       subroutine calling hierarchy with brief process outline
        C
        C COMMON Block Communication:
        C --------------------------
        C       /blockname/     descriptor              length (1*2 words)
        C                         host routine names
        C
        C           variable    (datatype)      descriptor
        C
        C               (These lists provide a module DATA DICTIONARY)
        C
        C order conventions or geometry definitions (if appropriate):
        C ---------------------------------------------------------

    Subroutine Headers:
    ---------- --------
        C
        C  Purpose:  brief process description
        C
        C  CALLed from:  calling routine name(s)
        C
        C  CALLs:  called routine name(s)
        C
        C  geometry definitions if appropriate:
        C----------------------------------------------------------

    On-line Flags:
    ------- ------

        Expression      Comment             Function
        ----------      -------             --------
           ...          ! Default       (default values set in USERx)
           ...          ! I*4           (four-byte integer arithmetic)
```

SECTION FIVE

TESTING

Verification of the TransMAPS package was divided into three major parts: the MAPS compression logic, the MAPS product formation options, and the annotation capability. This report of the TransMAPS development efforts concludes with a brief discussion of these tests.

5.1 Compression-Logic Diagnostic Test Image

The principal complexities in MAPS arise during the compression process and are further compounded by the extended space of user options in this step. This is the most likely area for small errors in logic or code entry. Moreover, because of the adaptive nature of MAPS, the effects of such errors might remain very localized and easily go undetected in review of the compression of a general image scene. Thus, tests of the compression logic must be capable of exhausting the various patterns of intensity which MAPS might encounter in real imagery. For this purpose, it is sufficient to include only 'generic' patterns which represent all geometries but not all possible intensity levels. A 'binary' image of light (gray scale 0) and dark (gray scale 255) will suffice.

Such a diagnostic test image was created for TransMAPS and is displayed as Figure 5-1. This image is 160 lines by 128 pixels and is suitable for the line-printer pseudo-image display mode of the TransMAPS ANNOTE module. An overlay pattern of lines along 'natural' MAPS boundaries has been added to the pseudo-image.

From the figure it is seen that four complete patterns representing 1x1, 2x2, 4x4, and 8x8 elements are displayed. Also, one quadrant of the pattern at 16x16 and a single dark element at 32x32 have been included. The full patterns at the four lowest resolutions are each made up of twelve 'quads' of light and dark elements. Indeed, all possible

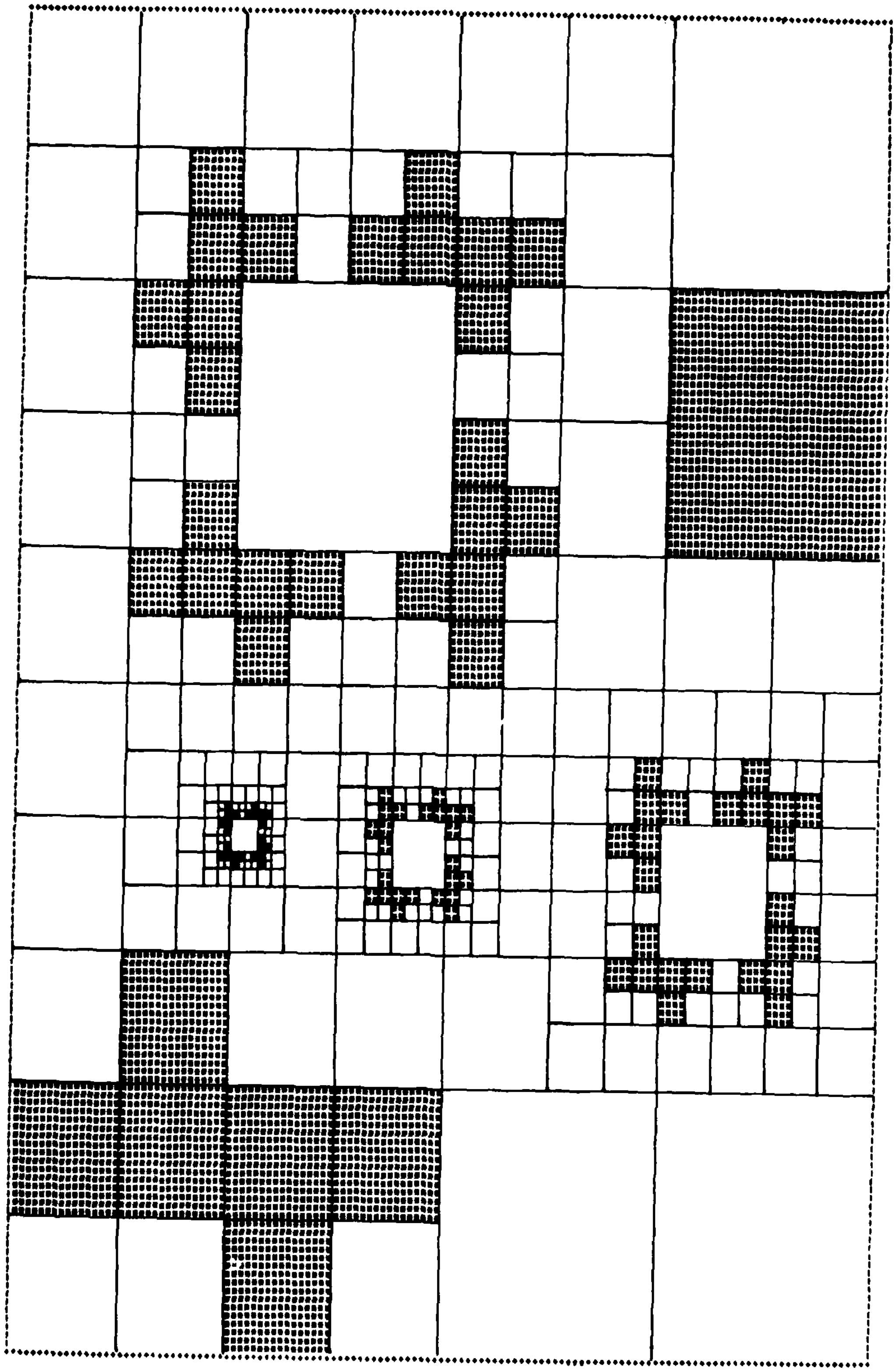


Figure 5-1. MAPS Compression-Logic Diagnostic Image

arrangements of 'one light and three dark', 'two adjacent light and two adjacent dark', and 'three light and one dark' are present for all four resolution levels. For the 16x16 case, at least one pattern from each of the three classes is represented. Note further that for each resolution, the quad patterns of 'all light' and 'all dark' are shown somewhere in the image.

This test image supports the following classes of verification diagnostics:

```
Compression-Logic Diagnostic Tests:
---------------- ---------- ------

        Image Partition and Macro-Fidelity Control:
        ----- --------- --- -------------- -------
                Input Image Line and Pixel Skips
                Subframe Phasing (Square and Staggered)
                Macro-Partition Group Assignment

        Micro-Fidelity Control:
        -------------- -------
                Zigzag Sequencing
                Contrast Control as a Function of Transition Level
                Contrast Control as a Function of Contrast Type
                Pattern Code Assignment

        Gray-Scale Manipulation:
        ---------- ------------
                Contrast Space Quad Sort
                Intensity Space Quad Sort
                Intensity Reset Assignment
```

Micro-performance of TransMAPS was verified in all of these areas. The diagnostic test image is also included as part of the TransMAPS package and can be used for verification on installation and, perhaps in an even more valuable role, to familiarize the user with the detailed effects of many of the user options. Several specific examples of its use are given in Section Eight of the User's Manual.

5.2 Product-Generation Familiarization Test Image

Another 'toy' image was generated by resampling the video-sized frame of the IEEE Girl image down to 120 lines by 128 pixels. The result is shown in the pseudo-image of Figure 5-2. The source version was also scaled to six-bit intensity (right-justified in eight bits) to test the six-bit option of the raster to subframe conversion. At this size, the image content is rather strongly undersampled but this makes it effective in emphasizing small artifacts in the resultant product images. The coarse gray-scale granularity of the pseudo-image also serves to enhance otherwise small artifacts. This combination, then, is very suitable for familiarizing the user with the types of fidelity compromises which the MAPS process introduces during compaction.

A series of products were generated to compare various compression and reconstruction strategies with the compression level held constant (at two bits per pixel). This entire sequence is displayed in Section Nine of the User's Manual. In addition to illustrating the strategies, the results of the sequence verify the expected performance characteristics of the various product generation options. This test image is also included in the TransMAPS package. Its small size makes it very effective in exploring a wide range of strategies at very modest computational investment.

5.3 Annotation Option Test Patterns

The various annotation options in ANNOTE were sampled with a small test pattern using the printer pseudo-image for display (see Section Four of the User's Manual). As the final test step, however, a much more complete test of the annotation capability was provided by constructing the test pattern shown in Figure 5-3. This result was generated on a PDP-11/70, transferred to magnetic tape, and then to film using an Optronics Photowrite. The test pattern shows all sixty symbols in all

Figure 5-2. MAPS Familiarization Test Image

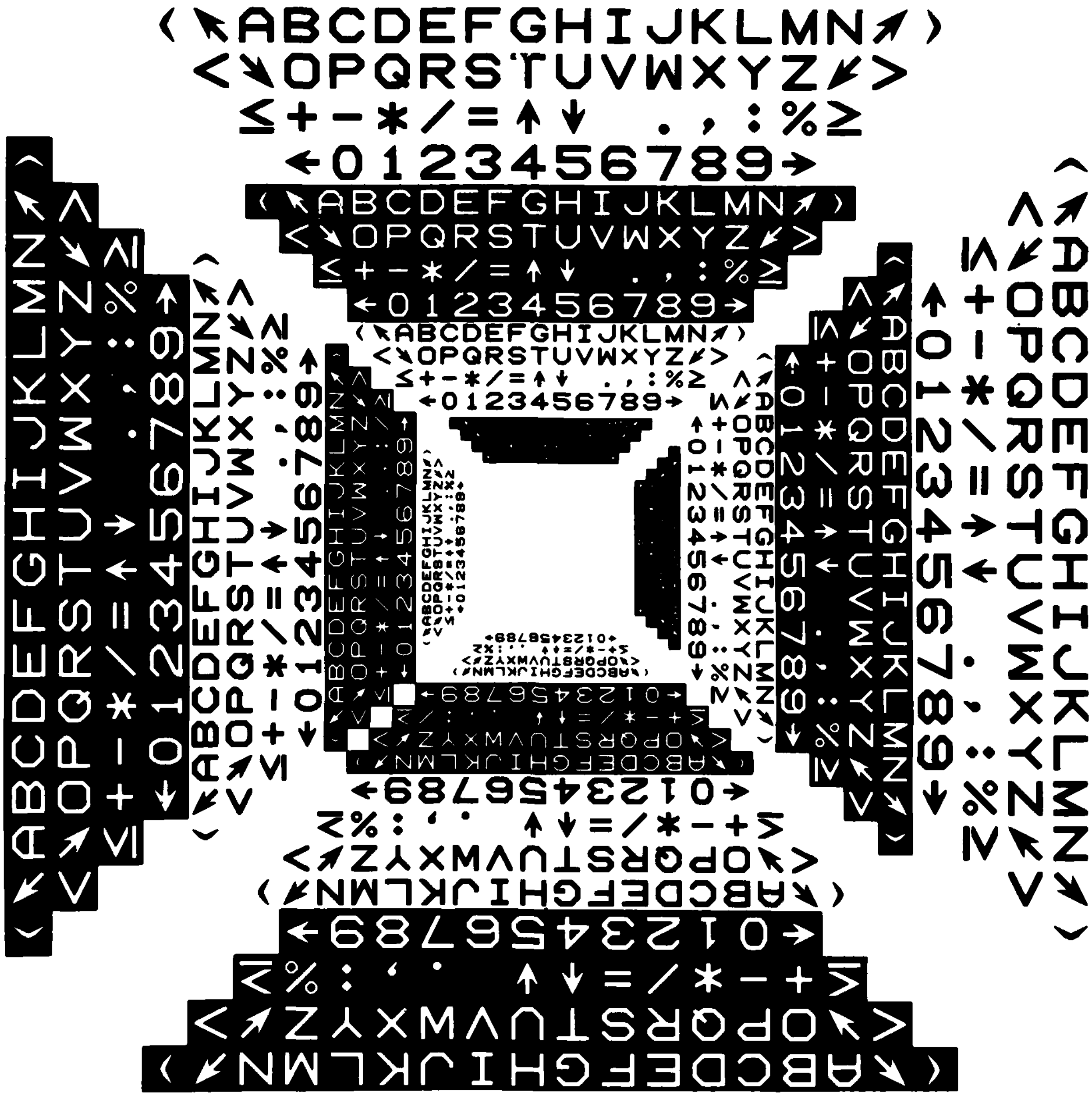


Figure 5-3. Annotation Options Test Pattern

four orientations at all four magnifications in alternate direct and complement presentations.

This concludes the brief discussion of the development of TransMAPS which constitutes this first part of the project "Final Technical Report." Indeed, this description is transitory and the 'scaffolding' it represents can be removed. The TransMAPS User's Manual is the central portion of this total document and is designed to stand alone in its support of the functional application of the TransMAPS package. Finally, the TransMAPS Maintenance Manual completes the current document. It presupposes understanding and familiarity with the User's Manual and adds the 'systems' perspective necessary to provide transparent software installation and maintenance to the user community.

PREFACE

This is Volume II of the Final Technical Report on Transportable MAPS Software. It constitutes the TransMAPS Software User's Manual; its companion volumes contain a description of TransMAPS development (Volume I) and the TransMAPS Maintenance Manual (Volume III). This volume is submitted in fulfillment of CDRL item A003 of Contract # F30602-80-C-0326.

TABLE OF CONTENTS

VOLUME II. TRANSMAPS USER'S MANUAL

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF TABLES

SECTION ONE

TRANSPORTABLE MAPS SOFTWARE: THE USER'S VIEW

This document provides formal description of the Transportable MAPS Software Package or 'TransMAPS' from a user's viewpoint.

1.1 Purpose and Applications

Micro-Adaptive Picture Sequencing (MAPS) is a computationally-efficient, contrast-adaptive, variable-resolution spatial image coding technique. The TransMAPS Software Package implements the MAPS processes and related support functions in an integrated software system which is designed to be transportable among a variety of high-use mini-computers in the DEC computer family. The purpose of this implementation is to broaden access to MAPS. The ultimate intent is to establish a vehicle suitable for direct exploration of the MAPS technique and to provide a system capable of supporting functional application of MAPS to real image coding tasks.

The current document addresses those areas specifically concerned with the user's application of the TransMAPS software. These include an overview of the structure of TransMAPS, general guidelines for user interaction, detailed information on user options, a concise description of the underlying MAPS concepts and processes, and several examples of MAPS interactive protocols. The intended audience includes both those who wish to use TransMAPS as a 'black box' for operational image coding and those who wish to explore the MAPS technique itself. The emphasis here is on 'what it does' and 'how to invoke it'.

1.2 User's Manual Organization

Sections Two through Five provide the information needed to gain an initial facility with the basic TransMAPS capability. Section Two presents an overview of the contents and structure of the package. Section Three discusses general guidelines for the TransMAPS interaction environment. Section Four then describes the stand-alone image assembly and annotation module, ANNOTE. This module provides rudimentary but rapid and effective image display support, even for systems with no formal image display device. Finally, Section Five describes the basic protocols needed for MAPS compression and decompression operations.

Sections Six through Nine cover topics intended to give the user much deeper understanding and fluency in applying TransMAPS. Section Six presents a summary of the underlying MAPS concepts and processes. Section Seven then describes the entire space of interactive user options in detail and on a module-by-module basis. Finally, Sections Eight and Nine present protocols for the extended use of the TransMAPS compression and product generation capabilities.

1.3 References

The information in this manual provides a self-contained guide to the use of TransMAPS. However, the related volumes - TransMAPS Final Technical Report and TransMAPS Maintenance Manual - may be helpful in giving additional context on the development of the package and its specific embodiment of the MAPS processes.

For more detailed information on the origin and evolution of MAPS, the user may also wish to consult the following reports and articles:

References:

RADC MAPS-Related Reports:
---- ------------ --------

LaBonte, A. E. and C. J. McCallum (Control Data Corporation), IMAGE COMPRESSION TECHNIQUES, RADC-TR-77-405, December 1977, Final Technical Report, Contract No. F30602-76-C-0350.

LaBonte, A. E. and T. E. Rosenthal (Control Data Corporation), MAPS IMAGE COMPRESSION, RADC-TR-80-173, May 1980, Final Technical Report, Contract No. F30602-78-C-0253.

LaBonte, A. E. (Control Data Corporation), INFRARED DATA COMPRESSION STUDY, RADC-TR-80-287, August 1980, Final Technical Report, Contract No. F30602-79-C-0080.

SPIE Proceedings Articles:
---- ----------- ---------

LaBonte, A. E., "Two-Dimensional Image Coding by Micro-Adaptive Picture Sequencing (MAPS)", Proceedings of the Society of Photo-Optical Instrumentation Engineers, Volume 119, APPLICATIONS OF DIGITAL IMAGE PROCESSING, pp 99-106, 1977.

LaBonte, A. E., "Micro-Adaptive Picture Sequencing (MAPS) in a Display Environment", Proceedings of the Society of Photo-Optical Instrumentation Engineers, Volume 249, ADVANCES IN IMAGE TRANSMISSION II, pp 61-70, 1980.

SECTION TWO

TRANSMAPS OVERVIEW

This section gives an overview of the contents and structure of TransMAPS.

2.1 The TransMAPS Package

The contents of the TransMAPS Package are summarized in Table 2-I. The installed software consists of seven process modules supported by an extended MAPS file structure and several related test image and data files.

The first six program modules are all directly MAPS-related. This group is further subdivided into modules concerned with MAPS compression (#1 and #2) and modules concerned with MAPS product formation (#3, #4, #5, and #6). The seventh module provides a stand-alone image assembly and annotation capability which also gives immediate image display support for the MAPS processes.

Modules #1 (SUBFRM) and #6 (RASTER) embody the basic image interface processes - conversion between external raster image format and internal MAPS subframe organization. Modules #2 (MAPS), #3 (DMAPS), #4 (ADAPT), and #5 (DIFFER) operate on subframe-organized imagery. They provide respectively, MAPS compression, MAPS decompression and resolution image formation, MAPS adaptive image smoothing, and MAPS difference image formation. Module #5 also yields an evaluation of MAPS fidelity performance. Finally, module #7 assembles up to two input raster images into a single frame and allows addition of annotation in a variety of orientations and type sizes. The resultant output image can be either in the form of another standard binary raster file or in the form of a formatted pseudo-image file suitable for listing on the system line printer.

TABLE 2-I. CONTENTS OF TRANSMAPS.

```
TransMAPS Package:
--------- --------

        Seven Main Program Modules:
        ----- ---- ------- --------
                #1  SUBFRM   Raster to Subframe Conversion      -  SF.FSV, SF.OBJ
                #2  MAPS     MAPS Compression                   -  MP.FSV, MP.OBJ
                #3  DMAPS    MAPS Decompression & Level Image Formation
                                                                -  DM.FSV, DM.OBJ
                #4  ADAPT    MAPS Adaptive Image Smoothing      -  AD.FSV, AD.OBJ
                #5  DIFFER   MAPS Difference Image Formation    -  DF.FSV, DF.OBJ
                #6  RASTER   Subframe to Raster Conversion      -  RS.FSV, RS.OBJ
                #7  ANNOTE   Image Assembly and Annotation      -  AI.FSV, AI.OBJ

                Provided on Computer-Compatible Tape (CCT):
                        FORTRAN IV-Plus Source Code  -  x.FSV
                        FORTRAN Object (F4P) Code  -  x.OBJ

        Six Data Sets:
        --- ---- -----
                Annotation Symbol-Map Tables                        -  SYMBOL.BIN
                MAPS Compression Test Image (160 x 128)             -  MTEST.BIN
                Sample MAPS User Parameter Set (Use with MTEST)     -  MSET.BIN
                MAPS Product Generation Test Image (120 x 128)      -  GIRL6.BIN
                Two 'Video Frame' Images (480 lines x 624 pixels):
                        Building Scene                              -  BLDGIMG.BIN
                        IEEE Girl                                   -  GIRLIMG.BIN

                Provided on Computer-Compatible Tape (CCT)

        MAPS File Structure with Standard Filenames and Headers:
        ---- ---- --------- ---- -------- --------- --- --------
                Source Image (One Subframe/FIXED Record)        -  IMAGE.DAT
                User Parameter Set                              -  MSET.DAT
                MAPS Compression Stream (FIXED Records)         -  MAPS.DAT
                MAPS Block/Pattern Image (Subframes)            -  DMAPS.DAT
                MAPS Resolution Image (Subframes)               -  LEVEL.DAT
                MAPS Adaptively Smoothed Image (Subframes)      -  ADAPT.DAT
                MAPS Difference Image (Subframes)               -  ERROR.DAT
                Fidelity Performance Summary (Listing)          -  EPRINT.DAT
                MAPS Product Image (Raster)                     -  xRAST.DAT
                                                                   x = I,D,L,A,E
                Annotation Symbol-Map Tables                    -  SYMBOL.DAT
                Annotated Image (Raster)                        -  ANIMG.DAT
                Annotated Printer Pseudo-Image (Listing)        -  APRINT.DAT

        User Interaction Pre-Planning Aids:
        ---- ----------- ------------ -----
                MAPS Planning Form  -  Compression
                MAPS Planning Form  -  Product Generation
                Annotation and Image Assembly Planning Form

        Documentation:
        --------------
                TransMAPS User's Manual
                TransMAPS Maintenance Manual
```

Six data sets are provided with the package. The first contains the bit-map tables for the annotation symbol set. This resides as a system data file, SYMBOL.DAT, and is read in automatically by module #7, ANNOTE when it is invoked. The second and third data sets consist of a 'toy'-sized diagnostic test image, MTEST.DAT, and a corresponding sample set of user parameters, MSET.DAT. The fourth data set is also a 'toy'-sized test image, GIRL6.DAT; it is particularly suitable for initial familiarization with the MAPS processes. The fifth and sixth data sets, BLDGIMG.DAT and GIRLIMG.DAT, are examples of real-world 'video'-sized images.

Communication of image data among the modules is provided through the MAPS file structure. Ancillary data is contained in standardized file headers and each file type has a unique but standard name. Thus, the intermodule data transfer is transparent to the user. However, the file names provide simple mnemonics for their contents so the user can assess the status of MAPS processing through a simple review of the appropriate file directory.

Application of TransMAPS is supported externally by a group of user aids and the set of software documentation. The aids are presented as three 'planning forms' for MAPS Compression, MAPS Product Generation, and Annotation and Image Assembly. These forms chart the extensive space of MAPS user options and allow its structure to be seen at a glance. User entries on the forms provide for both pre-planning and documentation of TransMAPS interactive sessions.

The formal software documentation consists of the TransMAPS Maintenance Manual and this TransMAPS User's Manual.

2.2 TransMAPS Structure

The structure of the TransMAPS software system is depicted in Figure 2-1. This presentation shows the detailed relationship between the seven TransMAPS process modules and the TransMAPS system of MAPS standard files. Figure 2-1 provides a self-contained roadmap to TransMAPS and should be viewed as the key reference whenever a system overview is required.

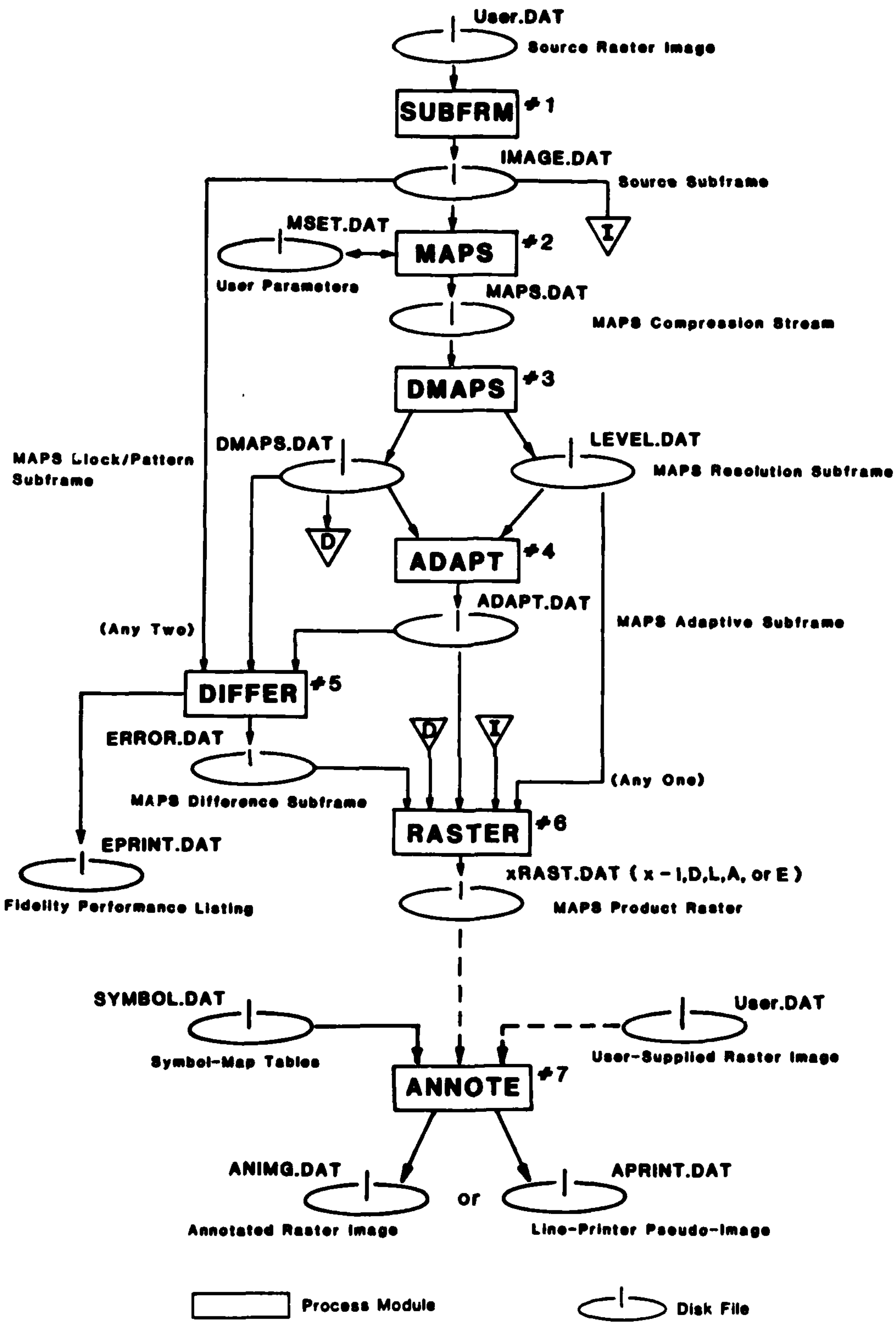


Figure 2-1. TransMAPS Structure

SECTION THREE

TRANSMAPS INTERACTION ENVIRONMENT

TransMAPS is designed to be extensively interactive to accommodate the large space of available user options. This section describes the general guidelines appropriate to these interactions.

3.1 Input and Output File Formats

The communication of external image data files to and from the TransMAPS system takes place in modules #1 (SUBFRM), #6 (RASTER) and #7 (ANNOTE). TransMAPS expects these images in the form of raster-organized binary files with one image line per logical record. The raster geometry conventions require the pixel index to increase sequentially along each line from left to right, and the line index to increase sequentially down the image from top to bottom. This is essentially a video convention for the geometry except that it does not involve line interlace.

Two pixel formats are supported for the external imagery in TransMAPS - an eight-bit pixel and six-bit pixel right-justified in an eight-bit field (the so-called DICOMED format).

TransMAPS is designed to accommodate logical records up to 4000 pixels (bytes) in length, although such large images are computationally expensive to process on a mini-computer system. However, in order to minimize buffer space requirements, TransMAPS expects the logical records to be partitioned into much smaller physical records through the use of the SEGMENTED record type. Since no universal standard for image file format is available, each local 'standard' digital image form is unknown a priori and likely to vary from site to site. Thus, responsibility for expressing the raster source imagery as a SEGMENTED binary file is

presumed to reside with personnel at each site. The test imagery supplied with the TransMAPS package is, of course, already in this SEGMENTED format.

3.2 Terminal Interaction Formats

Each of the seven TransMAPS modules is invoked by an MCR (DEC's Monitor Console Routine) command of the generic form 'RUN Taskname'. Here, 'Taskname' takes on either a long or short version for each module as listed in the following tabulation:

Module #1:	SUBFRM	or	SF
Module #2:	MAPS	or	MP
Module #3:	DMAPS	or	DM
Module #4:	ADAPT	or	AD
Module #5:	DIFFER	or	DF
Module #6:	RASTER	or	RS
Module #7:	ANNOTE	or	AI

Once the module is invoked, code internal to the module directs the interaction to select values or states for the required user options. Typically, the program requests an update to an option with a query in the form:

'Option descriptor' ? ('Allowed Range') 'Current Value'

The 'Allowed Range' is adjusted dynamically to account for any changes in constraint imposed by prior option selections. 'Current Value' initially displays the system default value for the option. Thereafter, it presents either the most recent selection or the closest value from the current allowed range.

Insofar as possible, user responses to these option queries are limited to a very small number of generic types. Consistent response formats have been sought. These types divide first into two general categories - numeric and alpha-literal. The numerics are mostly single numbers (a string of contiguous decimal digits, possibly with sign and decimal point). Occasionally, a numeric response in the form of a vector of numbers is required. In the vector case, successive components are separated by simple blank spaces. If several adjacent components have the same value, they may be entered with the sequence of a 'repeat count' followed by an asterisk followed by the common 'value' (r*v). For all numerics, a slash character (/) followed by a carriage return is used to denote the default response, 'no change'. A slash (/) inserted before a list of vector components is exhausted denotes that all remaining components are left unchanged.

The alpha-literal responses further subdivide into three types - Y or N for 'yes' or 'no'; a single mnemonic letter for a menu choice; and a contiguous symbol string for a filename, an image name, or a message text. The 'no change' default for alpha-literals is normally a simple carriage return. However, for consistency with the numeric case, a leading slash (/) is also interpreted as the 'no change' response. The only limitation that this imposes is that a message text cannot begin with a slash symbol (/).

Module #2 (MAPS) and module #7 (ANNOTE) have particularly extensive option spaces. In order to avoid the tedium of providing long strings of 'no change' responses, the interactions for these modules have been subdivided into smaller groups of related options. At the beginning of each group, the user is given the choice of entering the group for option review and update or simply accepting the current values for all options in the group.

The user is also given extensive recovery and change control in the interaction. For matrix option specifications such as those for macro-fidelity image partition or micro-fidelity contrast thresholds, the code provides immediate feedback by updating and displaying the matrix after each line (vector) of user responses. Moreover, the user can access any line again and in any order to make corrections. In effect, a simple 'editing' mode is available. Similarly, for extended text entry such as that which may be encountered in defining annotation messages, all text is collected and displayed together on the terminal screen after initial entry. Again, an 'editing' mode is enabled which allows individual messages from this collection to be repeatedly corrected until the total text is satisfactorily defined.

Finally, the last query in the interaction for each module gives the user the choice of returning to review (and possibly modify) the entire set of option selections or to proceed. This insures the opportunity for parameter verification before the image processing resources are committed.

3.3 PIP Filename Manipulations

The modular structure of TransMAPS allows many strategies and time sequences for the application of the package. For example, the raster to subframe conversion (SUBFRM) might be run once on a particular source image. Next, the MAPS compression process (MAPS) might be run several times with different control parameters before proceeding with MAPS product generation. Each invocation of a process module produces a new set of output files. For files with the same filename - for example, MAPS.DAT - successive files are distinguished by increasing file 'version numbers'. The file specification has the form MAPS.DAT;'version' where 'version' is an octal number. When a module is invoked which requires a

particular file type as input - for example, module #3 DMAPS requires MAPS.DAT - it automatically opens and accesses the latest version.

In order to access earlier versions of a file, some capabilities of DEC's PIP utility (Peripheral Interchange Program) can be exploited. In particular, the PIP switches '/RE', '/EN', and '/RM' along with the subswitch '/NV' are relevant. These switches play the following roles:

/RE	Rename - allows any portion of a file specification, 'filename.filetype;version' to be changed;
/EN	Enter - allows a synonym file specification to be entered into the file directory, access to the file is allowed under any of its synonyms;
/RM	Remove - allows a synonym to be removed from the file directory without removing the file or its other names;
/NV	New Version - supplies a 'version number' one larger than the latest version when used with /RE or /EN.

An example of the use of these switches is provided by the following generic command line:

```
PIP Filename.DAT/RE/NV = Filename.DAT;earlyversion.
```

In this case, 'earlyversion' is renamed to the (new) latest version and will be accessed by any module which opens 'Filename.DAT' for input.

Additional PIP facilities which are useful in controlling file proliferation are the purge and delete switches '/PU' and '/DE'. PURGE eliminates all but the most recent version associated with a particular filename and DELETE eliminates a file with an explicit version number specified. Obviously, these commands must be used with care to avoid inadvertent deletion of files which were to be kept.

These various PIP system utilities clearly provide great flexibility in maintaining the MAPS file environment.

SECTION FOUR

IMAGE ASSEMBLY AND ANNOTATION

Module #7, ANNOTE, provides a stand-alone image assembly and annotation capability which can be used either with MAPS products output from module #6 or with other user-supplied raster images. Because of its role in 'quick-look' pseudo-image display (using the system line printer), ANNOTE is introduced in this section, out of normal order. As a consequence, ANNOTE becomes available to provide visual output for examination of the results of MAPS process applications.

4.1 Annotation and Image Assembly Planning Form

The user options for ANNOTE are summarized in the 'Annotation and Image Assembly Planning Form' presented in reduced size as Figure 4-1. The actual form just fills the length of an 8-1/2" by 11" sheet - large enough for comfortable user entry. ANNOTE allows the assembly of up to two input raster images into a single output frame. Moreover, it can be run recursively to assemble several images by taking the output from a previous run as one of the inputs to the current run. Restriction to two input images at a time is largely a consequence of the 32K task size limit imposed by the sixteen-bit word length of the PDP-11.

ANNOTE has two output frame modes. In the first, an ordinary binary raster image file is created. This is the mode to be used for recursion or for making large image frames (up to 4000 pixels) for subsequent transfer to a large-format display device such as an Optronics Photowrite. The second mode is limited to image widths of 128 pixels per line and creates a formatted pseudo-image file to be listed on the system line printer. This mode has very coarse intensity granularity (only eight gray-scale levels) and uses only one overprint per line. Even with

ANNOTATION AND IMAGE ASSEMBLY PLANNING FORM

RUN ANNOTE

DATE ____________

OUTPUT IMAGE:

TYPE (RASTER FILE PRINTER LISTING) [R] LINES ________ PIXELS ________ COMPLEMENT (Y N) [N]
ANIMG.DAT APRINT.DAT

EMBEDDED IMAGES: NUMBER (0 1 2) [0]

IMAGE	FILENAME	INPUT POSITION SKIP LINES	INPUT POSITION SKIP PIXELS	SIZE LINES	SIZE PIXELS	OUTPUT POSITION START LINE	OUTPUT POSITION START PIXEL	COMPLEMENT
1	- - - - - - - - - -	[0]______	[0]______	______	______	______	______	(Y N)
2	- - - - - - - - - -	[0]______	[0]______	______	______	______	______	(Y N)

EMBEDDED MESSAGES: NUMBER (0 - 20) [0]

MSG	ORIENT	CHAR	SYMBOL SIZE	CENTER LINE	CENTER PIXEL	COMP	TEXT 1 5 10 15 20 25 30 35 40 45 50
1	(TBLR)	___	(1234)	___	___	(Y N)	- -
2	(TBLR)	___	(1234)	___	___	(Y N)	- -
3	(TBLR)	___	(1234)	___	___	(Y N)	- -
4	(TBLR)	___	(1234)	___	___	(Y N)	- -
5	(TBLR)	___	(1234)	___	___	(Y N)	- -
6	(TBLR)	___	(1234)	___	___	(Y N)	- -
7	(TBLR)	___	(1234)	___	___	(Y N)	- -
8	(TBLR)	___	(1234)	___	___	(Y N)	- -
9	(TBLR)	___	(1234)	___	___	(Y N)	- -
10	(TBLR)	___	(1234)	___	___	(Y N)	- -
11	(TBLR)	___	(1234)	___	___	(Y N)	- -
12	(TBLR)	___	(1234)	___	___	(Y N)	- -
13	(TBLR)	___	(1234)	___	___	(Y N)	- -
14	(TBLR)	___	(1234)	___	___	(Y N)	- -
15	(TBLR)	___	(1234)	___	___	(Y N)	- -
16	(TBLR)	___	(1234)	___	___	(Y N)	- -
17	(TBLR)	___	(1234)	___	___	(Y N)	- -
18	(TBLR)	___	(1234)	___	___	(Y N)	- -
19	(TBLR)	___	(1234)	___	___	(Y N)	- -
20	(TBLR)	___	(1234)	___	___	(Y N)	- -

Figure 4-1. Annotation and Image Assembly Planning Form

these severe dynamic range restrictions, the resultant 'images' convey a surprising amount of information. Indeed, certain small irregularities and artifacts are enhanced by the process so it makes a very effective tool for exploring MAPS fidelity performance.

Two of the main features of the ANNOTE option space have just been discussed. From the planning form in Figure 4-1, however, it is seen that there are three groups of options and each group contains several entries. These three categories will be discussed in the next three subsections.

4.2 Output Frame Specification

The first user option is the choice between output image types - raster file (the default) or printer pseudo-image. The standard filename for the raster file is ANIMG.DAT; the filename for the printer listing is APRINT.DAT.

The next pair of options specify the size of the output frame. Numeric values for the number of image LINES and the number of PIXELS per line are required here. The number of bits per pixel is always assumed to be eight in this module. Note that the 'video' geometry conventions - pixel index increasing left to right, line index increasing top to bottom - are in force for all images in ANNOTE.

The final output frame option is the choice of whether to make the frame background clear (gray-scale 0) or opaque (gray-scale 255). The default is 'clear' but a 'Y' response to the 'COMPLEMENT ?' query will change the background to 'opaque'.

4.3 Embedded Input Image Specifications

The first option is the choice of the number of input images to be assembled. The allowed values are 0, 1, and 2. A value of '0' skips the rest of the specification and would be chosen if a 'text-only' output frame were desired. A value of '1' would be chosen if it were desired to annotate a single image. A value of '2' implies the assembly of a pair of images, one or both of which might have been the output from previous ANNOTE invocations.

Each embedded input image then requires specification of eight additional selections. First, the filename for the input image must be entered; this can be up to nine characters long and is an alpha-numeric string. If the input image is a MAPS raster product, the filename will be one of IRAST, DRAST, LRAST, ARAST, or ERAST dependent on the product type. The latest version of the corresponding file will be accessed. Raster images other than MAPS products, or MAPS images which have been renamed using the PIP utilities, will have to have the corresponding filenames entered as they appear in the file directory. Note that in all cases, the file type must be '.DAT' and is not entered explicitly.

The next two input image parameters involve file positioning. ANNOTE contains provision for skipping into the image by a specified number of lines and pixels. The queries require numeric values for SKIP LINES and SKIP PIXELS. The default for each is zero.

The size of the embedded image is selected next through input of two numerics for LINES and PIXELS. Note that the embedded image must be smaller than the output frame; the program enforces this condition if

values which exceed this are transmitted. The number of pixels allowed is also constrained by the condition:

$$\text{PIXELS} + \text{SKIP PIXELS} < 4001$$

Two more numerics are required to specify the position of the embedded input image in the output frame. The position is given in terms of the output frame location for the START LINE and START PIXEL corresponding to the upper left pixel ('origin') of the input image. Here again, the allowed range is determined and displayed in terms of the relative sizes of the input and output frames; violations are automatically reset to the closest allowed value.

The final option for each input image is the choice of whether the gray scale is to be direct (the default) or COMPLEMENTED. This feature is useful in creating or converting images in 'negative' form.

Note that when two input images are to be assembled, they are allowed to 'conflict' within the output frame even though each must fit individually. The conflict resolution convention is that Input Image 2 overwrites Input Image 1 where they overlap. Thus, if successive images are being added to a frame by recursion, the output image from the previous ANNOTE run should be Input Image 1 in the current run.

4.4 Annotation Message Specifications

The number of annotation messages can range from zero to twenty and this is the first option selected in the embedded annotation interaction. If zero is chosen, the remainder of the interaction is skipped.

Each message, up to the number chosen, is then entered in a hierarchical fashion. Successive option choices for the message tend to constrain

later choices and this is reflected in the dynamic changes of the respective 'allowed range' displays. Seven options must be entered for each message.

Messages can be oriented in any of four directions with the top of the symbols toward the 'Top', 'Bottom', 'Left', or 'Right' of the output frame. This facility is provided so that the annotation can be matched to the scene content if the scene doesn't match the 'video' geometry. The user must respond with a single character literal T, B, L, or R to this orientation query.

Once the orientation has been specified, the number of symbols of each size which can be fit within the output frame can be determined. The next option requires the user to give a numeric specification of the message length in characters. This length can be up to the maximum allowed by the image frame at the desired symbol size (but no more than fifty characters if that is smaller). Allowed ranges as a function of symbol size are supplied as part of the interactive user prompt for this option selection. Note that a particular message can be deleted during any editing step by simply specifying a zero character count.

The symbol size is then chosen from those still allowed for this orientation and message length. Initially, symbol sizes of 1x, 2x, 3x, and 4x are possible. These correspond to characters in frames which are 16x16, 32x32, 48x48, and 64x64 pixels in size.

Location of the message in the output frame is selected next. This position is specified by giving the output frame CENTER LINE and CENTER PIXEL coordinates. Specification of the position of the center of the message avoids having to remember an orientation-dependent convention.

The user is also given the choice of whether the characters are to be

'direct' (opaque symbols in a clear frame) or COMPLEMENTED (clear symbols in an opaque frame).

Finally, the user supplies the message text stream itself. This is prompted with a display of the sixty allowed symbols in the annotation character set and a line of dashes corresponding to the chosen length of the message.

Note that annotation overwrites the embedded images where they overlap. Later messages can also overwrite earlier messages. However, this is usually not desired except in very special circumstances. Thus, ANNOTE provides a message conflict analysis and prompts editing of either the overwritten or overwriting message to correct the situation. Nevertheless, the conflict can be retained if desired.

4.5 Image Labeling Example

Figure 4-2 shows a completed planning form for a simple example of image labeling. The pseudo-image output mode has been chosen and a single input image is used. This image is the GIRL6.DAT 'toy' test image which has been converted from 6-bits to 8-bits by running it successively through SUBFRM and RASTER. Note also that the product image IRAST.DAT has been renamed to GIRL.DAT using PIP with a /RE switch.

The output frame was given twenty more lines than the input frame size. This extra space was then used to 'annotate' the input image with a message outside of its boundaries. The 'message' in this case simply lists the image size in 'lines x pixels'; the smallest symbol size was chosen here.

The results of the ANNOTE interactive session are displayed as Figure 4-3. Note here that the printer has been reset to

ANNOTATION AND IMAGE ASSEMBLY PLANNING FORM

RUN ANNOTE

DATE *March 1982*

OUTPUT IMAGE:

TYPE (RASTER FILE PRINTER LISTING) [N] LINES *140* PIXELS *128* COMPLEMENT (Y N) [N]
ANIMG.DAT APRINT.DAT

EMBEDDED IMAGES: NUMBER (0 1 2) [0]

IMAGE	FILENAME	INPUT POSITION SKIP LINES	INPUT POSITION SKIP PIXELS	SIZE LINES	SIZE PIXELS	OUTPUT POSITION START LINE	OUTPUT POSITION START PIXEL	COMPLEMENT
1	*GIRL*	[0] *0*	[0] *0*	*120*	*128*	*1*	*1*	(Y N)
2		[0]	[0]					(Y N)

EMBEDDED MESSAGES: NUMBER (0 - 20) [0] *1*

MSG	ORIENT	CHAR	SYMBOL SIZE	CENTER LINE	CENTER PIXEL	COMP	TEXT 1 5 10 15 20 25 30 35 40 45 50
1	(TBLR)	*7*	(1234)	*130*	*64*	(Y N)	*120x128*
2	(TBLR)		(1234)			(Y N)	
3	(TBLR)		(1234)			(Y N)	
4	(TBLR)		(1234)			(Y N)	
5	(TBLR)		(1234)			(Y N)	
6	(TBLR)		(1234)			(Y N)	
7	(TBLR)		(1234)			(Y N)	
8	(TBLR)		(1234)			(Y N)	
9	(TBLR)		(1234)			(Y N)	
10	(TBLR)		(1234)			(Y N)	
11	(TBLR)		(1234)			(Y N)	
12	(TBLR)		(1234)			(Y N)	
13	(TBLR)		(1234)			(Y N)	
14	(TBLR)		(1234)			(Y N)	
15	(TBLR)		(1234)			(Y N)	
16	(TBLR)		(1234)			(Y N)	
17	(TBLR)		(1234)			(Y N)	
18	(TBLR)		(1234)			(Y N)	
19	(TBLR)		(1234)			(Y N)	
20	(TBLR)		(1234)			(Y N)	

Figure 4-2. Image Labeling Planning Form

eight-lines-per-inch mode to make the 'pixels' more nearly square (a ratio of 8:10 rather the normal printer 6:10). This printer mode is recommended for consistent use with ANNOTE.

4.6 Annotation Options Examples

Figure 4-4 presents the completed planning form for an 'annotation-only' output frame which exhibits samples of the various annotation options. Again, the pseudo-image output form was selected but this time with the background COMPLEMENTED. The messages were constructed to show the range of symbol size, the four message orientations, direct and COMPLEMENTED messages, and the complete set of sixty characters.

The resultant printer listing is displayed as Figure 4-5. The various messages fill the output frame except for a small rectangular patch in the upper left-hand corner; there the effect of the background complement is seen. Independent planning and replication of this image is an excellent exercise to familiarize the new user with the annotation process.

A much more ambitious illustration of the annotation options is portrayed in Figure 4-6. Here, all sixty symbols appear in all four sizes at all four orientations with alternate direct and COMPLEMENT intensities. This example was prepared on a PDP-11/70 and then transferred via magnetic tape to an off-line Optronics Photowrite facility. Figure 4-6 summarizes how the ANNOTE annotation options appear in a real image display mode.

Figure 4-3. Example of ANNOTE Image Labeling

<u>Annotation and Image Assembly Planning Form</u>

RUN ANNOTE Date *March 1982*

Output Image:

Type (Raster file Printer listing) [R] Lines *176* Pixels *128* Complement (Y N) [N]

ANIMG.DAT APRINT.DAT

Embedded Images: Number (0 1 2) [0]

Image	Filename	Input Position Skip Lines	Input Position Skip Pixels	Size Lines	Size Pixels	Output Position Start Line	Output Position Start Pixel	Complement
1	----------	[0]	[0]					(Y N)
2	----------	[0]	[0]					(Y N)

Embedded Messages: Number (0 - 20) [0] *18*

MSG	Orient	Char	Symbol Size	Center Line	Center Pixel	Comp	Text (1 … 50)
1	(TBLR)	2	(1234)	24	16	(Y N)	1^
2	(TBLR)	1	(1234)	48	16	(Y N)	2
3	(TBLR)	1	(1234)	88	24	(Y N)	3
4	(TBLR)	1	(1234)	144	32	(Y N)	4
5	(TBLR)	5	(1234)	8	88	(Y N)	ABCDE
6	(TBLR)	5	(1234)	56	120	(Y N)	FGHIJ
7	(TBLR)	5	(1234)	104	88	(Y N)	KLMNO
8	(TBLR)	5	(1234)	56	56	(Y N)	PQRST
9	(TBLR)	3	(1234)	24	88	(Y N)	UVW
10	(TBLR)	3	(1234)	88	88	(Y N)	XYZ
11	(TBLR)	4	(1234)	32	40	(Y N)	÷⊥∶%
12	(TBLR)	3	(1234)	40	88	(Y N)	↘↑↗
13	(TBLR)	3	(1234)	56	88	(Y N)	←≡→
14	(TBLR)	3	(1234)	72	88	(Y N)	↙↓↘
15	(TBLR)	4	(1234)	120	96	(Y N)	±∠=≈
16	(TBLR)	4	(1234)	136	96	(Y N)	<≤≥>
17	(TBLR)	4	(1234)	152	96	(Y N)	5678
18	(TBLR)	4	(1234)	168	96	(Y N)	(90)
19	(TBLR)		(1234)			(Y N)	
20	(TBLR)		(1234)			(Y N)	

Figure 4-4. Annotation Example Planning Form

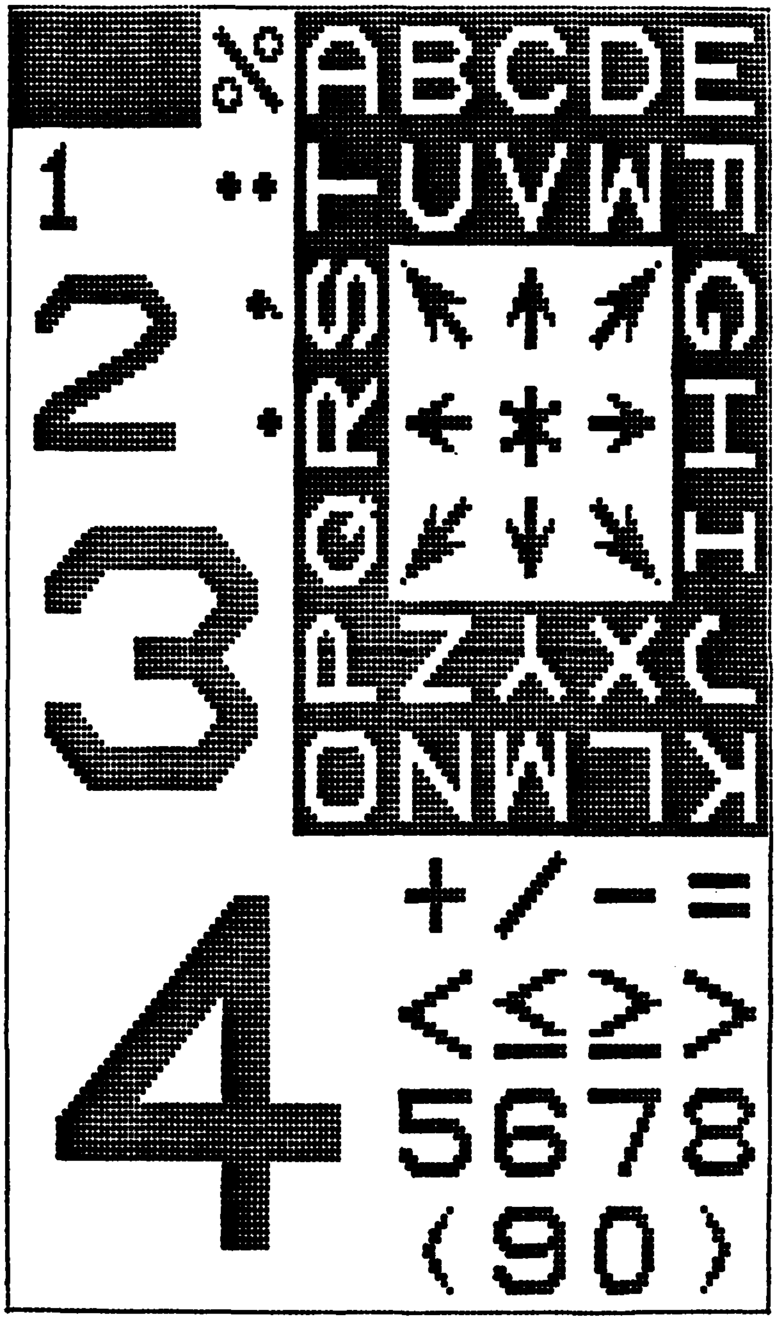


Figure 4-5. Pseudo-image Annotation Examples

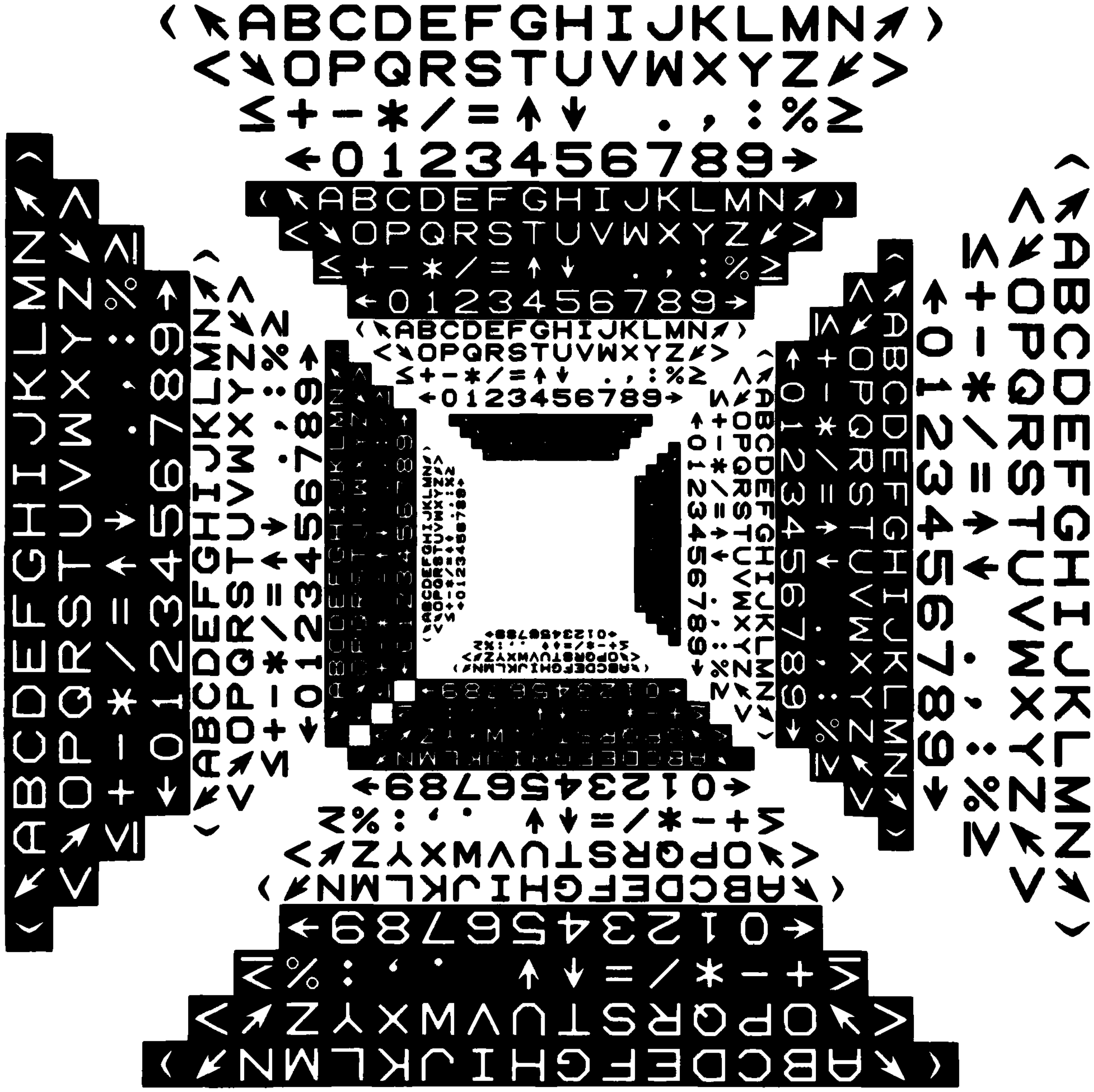


Figure 4-6. Annotation Options

SECTION FIVE

MAPS COMPRESSION/DECOMPRESSION: BASIC USE

The basic or core MAPS processes are contained in four modules - #1 SUBFRM, #2 MAPS, #3 DMAPS, and #6 RASTER. This section describes the broad classes of user interaction required by each of these modules. It then illustrates the entire flow by presenting the complete interactive protocol for a simple example. The protocol is a direct photocopy of the resulting DEC-writer listing.

5.1 Raster to Subframe Conversion - SUBFRM

The user interaction in module #1 is divided into four option groups. The first group involves 'source image indentification' and requires specification of the filename for the source raster and a user image name to be carried in the MAPS standard file headers.

The second group involves 'source image position specification' and requires input of the number of lines and pixels to be skipped into the source raster file. The defaults are zero for both lines and pixels.

The third group involves 'source image size specification' and requires input of the number of lines, the numbr of pixels per line, and the number of bits per pixel for the portion of the source raster frame to be processed. The number of pixels to be skipped plus the number of pixels retained can total up to 4000. The number of bits per pixel can be either eight or six right-justified in eight; the default is eight.

The final group involves 'source image partition specification' and requires choice of the subframe size to be used. The subframes can be 8x8, 16x16, or 32x32 pixels. The source image frame is automatically

padded in both the line and pixel directions to allow division into an integral number of complete subframes. If a subframe size of 8x8 is chosen, the user has the additional option of selecting between a square grid of subframes or a 'staggered' grid. In the staggered case, each successive subframe along the pixel direction starts three lines later (or five lines earlier) than the immediately preceding subframe. This gives the grid somewhat of a 'brick wall' appearance and tends to break up the 'blockiness' of the partition. The default selections are 8x8 subframes in a square grid.

5.2 MAPS Compression - MAPS

The complete option space for module #2 allows subtle and flexible control over the MAPS compression process. However, for basic MAPS compression applications, a very simple control strategy suffices. The first query in the interaction for this module requires selection of the subsequent interactive mode. For routine use, the 'Quick' mode should be selected. The other two modes, 'User' and 'Full', will be elaborated in Section Seven. Note that the 'Quick' mode is also the default.

The only other option required of the user, then, is specification of the 'Contrast Scale' for the MAPS micro-control thresholds. This is the one parameter which is strongly dependent on scene content and overall image intensity statistics. An intuitive sense of approximate range for a given image type and desired compression should emerge with experience. Initially, however, empirical exploration appears to be required here.

5.3 MAPS Decompression - DMAPS

This module generates the tonal decompressed image from the MAPS data stream. It also creates a companion MAPS 'resolution' image from this same stream but it is an ancillary product and will be illustrated

later. MAPS decompression is automatic and requires no interaction other than initial invocation of the module.

5.4 Subframe to Raster Conversion - RASTER

The only option to be specified in this module is the selection of the MAPS product type. Two products are relevant for the basic application of MAPS. The first is simply the re-establishment of the source image in an eight bit version and raster format with only the selected number of lines and pixels retained. Here, the user selects conversion of file IMAGE.DAT to file IRAST.DAT.

The other, more interesting, basic product is the MAPS decompressed tonal image. Here, the user selects conversion of file DMAPS.DAT to DRAST.DAT.

5.5 User Interaction Protocol ('Quick' Mode)

The following sequence of modules was invoked to illustrate the basic application of MAPS to *image coding and reconstruction:*

RUN	SF	Conversion of source image to subframes
RUN	MP	MAPS compression
RUN	DM	MAPS decompression
RUN	RS	Conversion of source image to raster
RUN	RS	Conversion of MAPS image to raster
RUN	AI	Assembly of images for printer display

The input image was chosen as the GIRL6.DAT 'toy' image and an 8x8 staggered subframe partition was used.

The SUBFRM protocol follows and is just as it evolved on the DEC-writer. User responses are left-justified along the edge of the listing. The listing is essentially self-descriptive:

```
 RUN DR1:[50,27]SF

********************************************
*                                          *
* MAPS RASTER TO SUBFRAME CONVERSION MODULE *
*                                          *
********************************************

  SOURCE IDENTIFICATION:

    SOURCE RASTER FILENAME? (UP TO 9 CHARACTERS)  FOR002
GIRL6

    USER IMAGE NAME? (UP TO 8 CHARACTERS)
GIRL

  SOURCE IMAGE POSITION:

    NUMBER OF LINES TO SKIP?    0     (/ = NO CHNG;
/
    NUMBER OF PIXELS TO SKIP? (< 4000)    0    (/ = NO CHNG)
/

  SOURCE IMAGE SIZE:

    NUMBER OF LINES TO PROCESS?  480  (/ = NO CHNG)
120
    NUMBER OF PIXELS TO PROCESS? (UP TO 4000)  624  (/ = NO CHNG)
128
    NUMBER OF BITS/PIXEL? (6 8)  8    (/ = NO CHNG)
6

  SOURCE IMAGE PARTITION:

    SUBFRAME EDGE? (8 16 32)  8       (/ = NO CHNG)
/
    STAGGER GRID? (Y OR N) N
Y


  USER SPECIFICATION COMPLETE:
  ****************************

    REVIEW? (Y OR N)  N
N

CONVERTING IMAGE GIRL     TO    254 SUBFRAMES
>
```

The MAPS protocol shows selection of the 'Quick' mode and the subsequent choice of a 'Contrast Scale' of 72. The 'SAVE ...' query is associated with retention of parameters for later use in the 'User' mode; it will be discussed in Section Eight. The resulting DEC-writer listing is:

```
  RUN DR1:[50,27]MP
 ***************************
 *                         *
 * MAPS COMPRESSION MODULE *
 *                         *
 ***************************

   USER OPTION MODES:

     Q - QUICK MODE (SELECT CONTRAST SCALE ONLY)
     U - USER PRE-DEFINED PARAMETERS FROM FILE MSET.DAT
     F - FULL OPTION REVIEW AND SELECTIVE REVISION

     MODE? (Q  U  F)  Q
 Q

     CONTRAST SCALE?   20.0     (/ = NO CHNG)
 72

   USER SPECIFICATION COMPLETE:
   ****************************

     REVIEW? (Y OR N)  N
 N
   SAVE THESE PARAMETERS FOR FUTURE USE? (Y OR N) N
 N

 MAPS COMPRESSING IMAGE GIRL     ,  120 LINES BY  128 PIXELS

 MAPS FILE CONTAINS     7 512-BYTE RECORDS PLUS 413 BYTES IN THE LAST

 MAPSEL DISTRIBUTION:

   LEVEL:      0         1         2         3
   COUNT:     1172      1571      422        32

   OPTIMAL BIAS: -       -6        -2        0
                 +        9         5        0

 COMPRESSION RATIO:   2.882 : 1

 BITS/PIXEL: 2.08177

 MEAN SQUARE ERROR:  0.17221 %
 >
```

Note that after completion of the compression task, module #2 returns a brief summary of results. These include: the number of 512-byte records required for the MAPS stream; the distribution of 'MAPSels' by size; optimal bias values for the subsequent pattern decompression (see Section Six); the compression level; and an overall fidelity measure in the form of the mean square error (MSE) in percent.

The protocol for DMAPS is very short:

```
RUN DR1:[50,27]DM
*********************************************
*                                           *
* MAPS DECOMPRESSION/RESOLUTION IMAGE MODULE *
*                                           *
*********************************************
  NO USER INPUTS REQUIRED

MAPS DECOMPRESSING IMAGE GIRL     ,  120 LINES BY  128 PIXELS
>
```

The protocols for the two subframe-to-raster conversion runs are also self-explanatory. Note that RASTER reports both the user image name and the type of product being formed:

```
 RUN DR1:[50,27]RS
********************************************
*                                          *
* MAPS SUBFRAME TO RASTER CONVERSION MODULE *
*                                          *
********************************************

  MAPS PRODUCT IMAGE TYPE:

      I - IMAGE (ORIGINAL SOURCE)
      D - DMAPS (MAPS DECOMPRESSED)
      L - LEVEL (MAPS RESOLUTION CODES)
      A - ADAPT (ADAPTIVELY SMOOTHED)
      E - ERROR (DIFFERENCE)

    TYPE? (I D L A E)  D[MAPS]
I

  USER SPECIFICATION COMPLETE:
  ****************************

    REVIEW? (Y OR N)  N
N

CONVERTING IMAGE GIRL     , FILE TYPE: IMAGE

  TO  120 LINE BY  128 PIXEL RASTER, FILE TYPE: IRAST
>
```

```
RUN DR1:[50,27]RS

********************************************
*                                          *
* MAPS SUBFRAME TO RASTER CONVERSION MODULE *
*                                          *
********************************************


  MAPS PRODUCT IMAGE TYPE:

      I - IMAGE (ORIGINAL SOURCE)
      D - DMAPS (MAPS DECOMPRESSED)
      L - LEVEL (MAPS RESOLUTION CODES)
      A - ADAPT (ADAPTIVELY SMOOTHED)
      E - ERROR (DIFFERENCE)

    TYPE? (I D L A E)  D[MAPS]
D

  USER SPECIFICATION COMPLETE:
  ****************************

    REVIEW? (Y OR N)  N
N

CONVERTING IMAGE GIRL    , FILE TYPE: DMAPS

  TO  120 LINE BY  128 PIXEL RASTER, FILE TYPE: DRAST
>
```

Finally, the protocol for ANNOTE image assembly follows:

```
RUN DR1:[50,27]AI

****************************************
*                                      *
* IMAGE ASSEMBLY AND ANNOTATION MODULE *
*                                      *
****************************************


  OUTPUT IMAGE SPECIFICATION:

    OUTPUT FILE MODE:

       R  -  GRAY SCALE RASTER IMAGE FILE 'ANIMG.DAT'
       P  -  LINE PRINTER PSEUDO IMAGE FILE 'APRINT.DAT'

    MODE?  (R P)  R
P

    NUMBER OF LINES?  800      (/ = NO CHNG)
241
    NUMBER OF PIXELS? (UP TO  128)  128    (/ = NO CHNG)
/
    COMPLEMENT BACKGROUND? (Y OR N)  N
```

```
  EMBEDDED IMAGES:

    NUMBER OF IMAGES? (0 1 2)  0      (/ = NO CHNG)
2
    IMAGE 1:

      FILENAME? (UP TO 9 CHARACTERS)  FOR002
IRAST
      SKIP LINES INTO INPUT IMAGE?    0      (/ = NO CHNG)
/
      SKIP PIXELS INTO INPUT IMAGE? (<4000)    0      (/ = NO CHNG)
/
      NUMBER OF LINES? (UP TO  241)   241     (/ = NO CHNG)
120
      NUMBER OF PIXELS? (UP TP  128)   128     (/ = NO CHNG)
/
      STARTING LINE? (RANGE 1 - 122)     1     (/ = NO CHNG)
/
      STARTING PIXEL? (RANGE 1 -   1)     1     (/ = NO CHNG)
/
      COMPLEMENT IMAGE? (Y OR N)  N
N
    IMAGE 2:

      FILENAME? (UP TO 9 CHARACTERS)  FOR003
DRAST
      SKIP LINES INTO INPUT IMAGE?    0      (/ = NO CHNG)
/
      SKIP PIXELS INTO INPUT IMAGE? (<4000)    0      (/ = NO CHNG)
/
      NUMBER OF LINES? (UP TO  241)   241     (/ = NO CHNG)
120
      NUMBER OF PIXELS? (UP TP  128)   128     (/ = NO CHNG)
/
      STARTING LINE? (RANGE 1 - 122)     1     (/ = NO CHNG)
122
      STARTING PIXEL? (RANGE 1 -   1)     1     (/ = NO CHNG)
/
      COMPLEMENT IMAGE? (Y OR N)  N
N

  EMBEDDED ANNOTATION:

    NUMBER OF MESSAGES? (0 -  20)   0     (/ = NO CHNG)
/

  USER SPECIFICATION COMPLETE!
  ****************************

    REVIEW? (Y OR N)  N
N

ASSEMBLING AND ANNOTATING IMAGE:

      241 LINES BY  128 PIXELS TO FILE 'APRINT.DAT'
>
```

The resulting pseudo-image is displayed in Figure 5-1.

Figure 5-1. Basic MAPS Compression Example.
Top - original; Bottom - MAPS Decompression

SECTION SIX

MAPS CONCEPTS AND PROCESSES

This section provides a synopsis of the key MAPS processes and concepts. More detailed discussions can be found in the references listed in Section 1.3

6.1 Image Partitioning

The MAPS partition of an image from the full frame down to the level of individual pixels is a two-stage process. First the frame is divided into square subframes, all of the same size. The pixel count for the subframe edge is required to be a power of two. Subframe sizes of 8x8 pixels, 16x16 pixels, or 32x32 pixels are allowed.

The subframes tesselate the image in either a square grid or a grid which is 'staggered' in one direction to give a 'brick wall' effect. Stagger is allowed only with the 8x8 subframe size and is intended to break up the perceptible 'blockiness' of the grid.

Within each subframe, the image is further divided by successive 'quartering'. This results in a series of nested 'quads', each quad having an edge pixel count which is a power of two. This division continues until the original pixel size is reached.

MAPS recodes the image from many simple fixed-sized pixels into a variable resolution pattern based on the image content. Each MAPS element or 'MAPSel' coincides with one of the natural quad units. MAPSels can range in size from original pixels up to entire subframes. The MAPSels are constrained, however, to give a complete (non-overlapping) tesselation of the image.

Thus, the MAPS partition processes involve a 'gridding' operation to form subframes, and a 'quadtree' division within the subframes. These concepts and associated labeling conventions are summarized in Figure 6-1.

6.2 Sequence Conventions

MAPS coding involves order conventions for sequencing the subframes and for sequencing elements within the subframes which allows element position information to remain implicit. That is, the element position is given by the location of the element in the storage sequence.

Square subframes are ordered in a simple coarse raster proceeding through rows of subframes in the 'pixel' direction and then advancing from row to row in the 'line' direction. Staggered subframes are ordered first by their startline and then by their position in the pixel direction. Thus, every eighth subframe along the pixel direction is given in sequence, and the startline is then advanced to the next row of every eighth subframe (see the stagger pattern in Figure 6-1).

Within each subframe, the nested quad pattern is traced from the lowest composite index to the highest. This results in the 'zig-zag' pattern through the source image as shown in the upper portion of Figure 6-2. This zig-zag pattern is also applicable to a valid sequence of MAPSels if the subpatterns within each larger element are collapsed to a point. An example of the resulting MAPS sequence is displayed in the bottom portion of Figure 6-2.

In essence, the MAPS order convention is that of a 'sequential quadtree'.

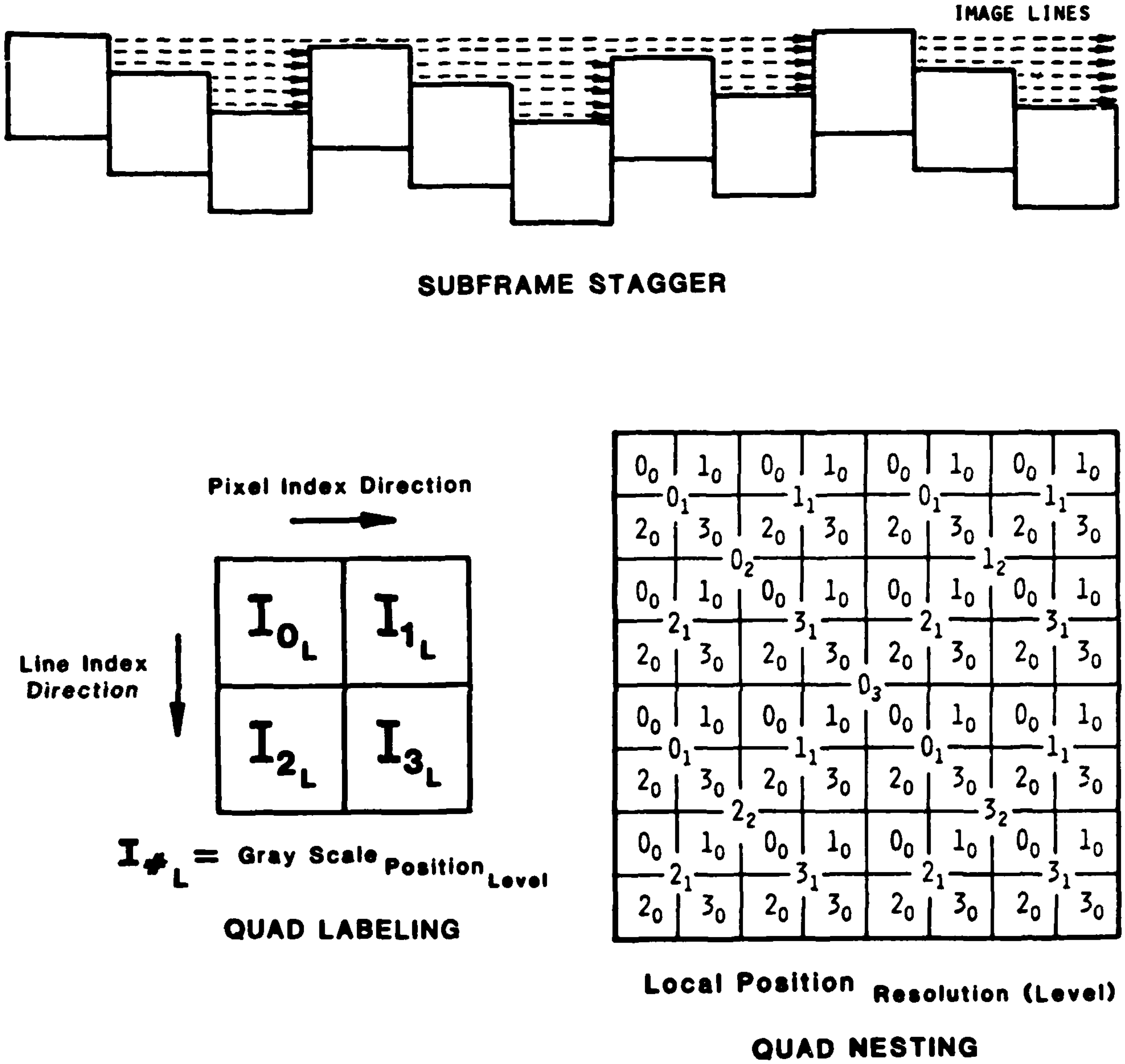
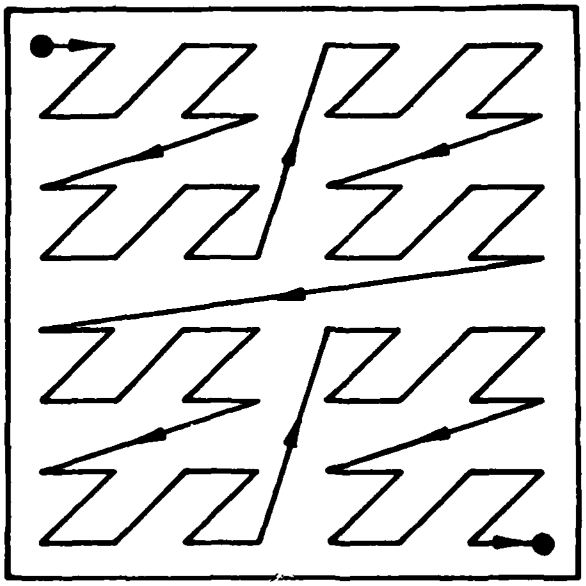
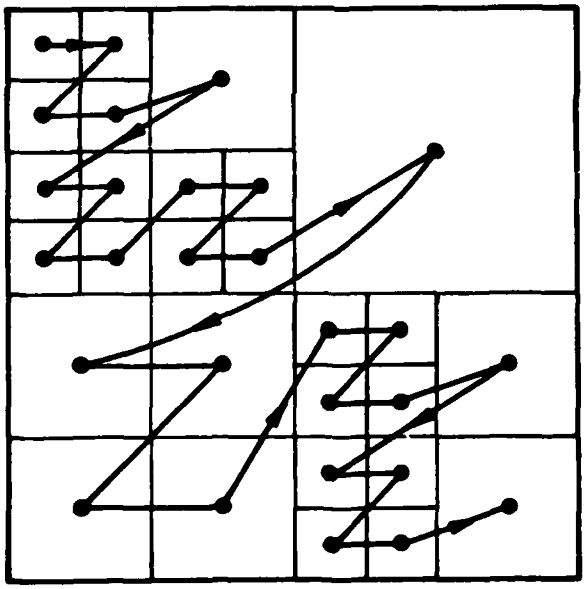


Figure 6-1. MAPS Image Partition Concepts



SOURCE SEQUENCE



MAPS SEQUENCE (Example)

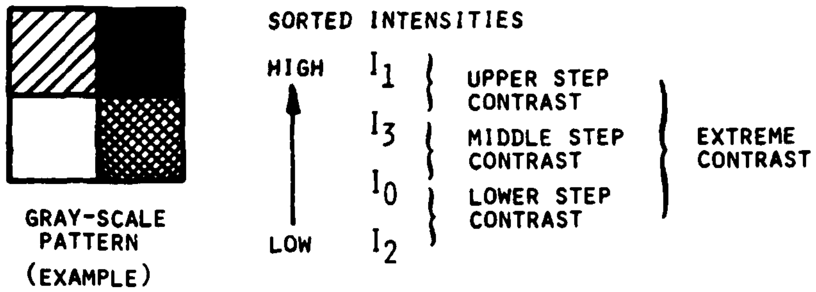
Figure 6-2. MAPS Sequence Concepts

6.3 Micro-fidelity Control

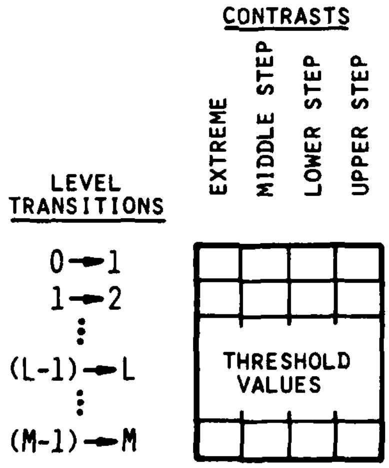
Formation of MAPSels larger than the original pixels involves successive evaluation of quad 'contrast'; testing of these contrasts against control thresholds; and composition of a quad into the next larger single element if no thresholds are exceeded. For each 'level' in the nest of quads, four contrasts are defined among the quad components as shown in the upper portion of Figure 6-3. A separate threshold is applied against each of these contrast types. This allows control on both adjacent intensity steps among the four elements and on the overall intensity range within the quad. The thresholds depend not only on the particular contrast type, but also on the level (element size) of the quad. This leads to a matrix of contrast control thresholds as depicted in the lower half of Figure 6-3.

In general, the threshold should be smaller for the 'step' contrasts than for the 'extreme' contrast. Moreover, the 'middle' step threshold may be set smaller than the 'outer' steps to preserve patterns where faint horizontal or vertical edges coincide with the quad centerlines. Finally, the thresholds should decrease rapidly with increasing element size since small intensity differences are much more noticeable among larger blocks. These observations are summarized in the plot given as the upper portion of Figure 6-4.

The contrast threshold matrix may be set directly by selecting each of its components. However, the threshold loci suggested in Figure 6-4 can be generated from a smaller set of parameters. Specification from a four-parameter set is illustrated in the lower part of Figure 6-4. the first parameter is an overall 'contrast scale' which is the 'extreme' threshold for the level transition from 1x1 to 2x2 elements. The 'recursive taper base' provides exponential threshold decay with increasing level. The 'step fraction' specifies the 'middle step' in

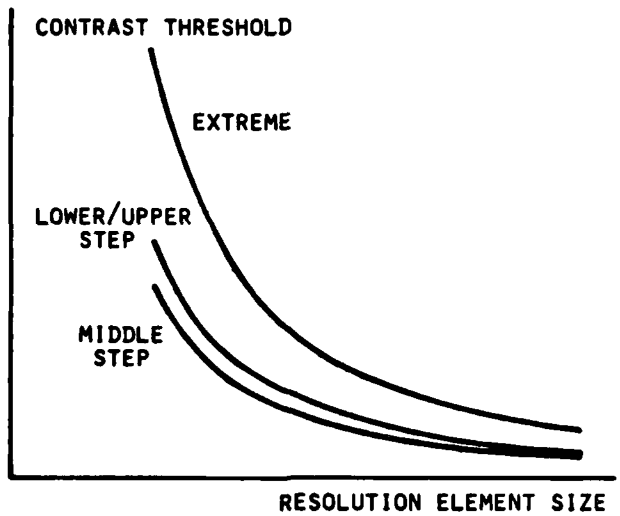


CONTRAST DEFINITIONS



CONTRAST CONTROL MATRIX

Figure 6-3. MAPS Contrast Control



CONTRAST THRESHOLD LOCI

$\text{Extreme}\ (L \rightarrow L+1) = \text{Extreme}\ (L-1 \rightarrow L)\ /\ B$

$\text{Middle}\ (L \rightarrow L+1) = F * \text{Extreme}\ (L \rightarrow L+1)$

$\text{Lower}\ (L \rightarrow L+1) = (F + \Delta) * \text{Extreme}\ (L \rightarrow L+1)$

$\text{Upper}\ (L \rightarrow L+1) = (F + \Delta) * \text{Extreme}\ (L \rightarrow L+1)$

WHERE USER INPUTS SPECIFY:

Extreme (0 → 1) - Contrast Scale
B - Recursive Taper Base
F - Step Fraction
Δ - Step Bias

PARAMETRIC MATRIX DEFINITION

Figure 6-4. MAPS Threshold Selection

terms of the 'extreme' threshold at a given level. Finally, the 'step bias' makes the 'outer' step thresholds larger than that for the 'middle step' at the same level.

Extensive empirical studies have shown that 'universal' values can be chosen for the taper base, step fraction, and step bias with little loss of performance over a wide range of imagery. These values are:

Taper Base, B	3.0
Step Fraction, F	0.5
Step Bias, Δ	0.1.

The contrast scale, however, is strongly dependent on image content and intensity distribution. Thus, this is the one control parameter which must be chosen in the 'Quick' mode of MAPS option selection.

6.4 MAPSel Coding

MAPSels are increased in size until some *quad threshold is exceeded*, or until not all four sub-components of the quad are available (due to a prior threshold violation). The resulting sequence of MAPSels must then be coded in such a way that the image scene can be reconstructed.

As has already been discussed, position information remains implicit in the MAPSel stream. However, intensity anbd resolution information are given explicitly. For 8x8 subframes, there are only four allowed MAPSel levels so the resolution code occupies two bits per MAPSel. For compatibility with typical machine environments, such MAPSels are taken in groups of four and the four two-bit resolution codes are packed into a single byte. This is then followed by four bytes of intensity information.

For 16x16 and 32x32 subframes, there are five and six possible states, respectively. In this case, the MAPSels are taken in groups of three. The 16x16 subframe case requires 5x5x5 = 125 states to describe the resolution code triplet. The 32x32 case requires 6x6x6 = 216 states. Both of these fit within a single byte. Indeed, there is a bit 'left over' in the 16x16 case and this can be used for internal parity if desired. Again, the resolution code byte is followed by the corresponding intensity bytes (three).

Two forms of intensity coding are used. In the 'block' mode, a uniform intensity over the entire MAPSel is coded as an eight-bit byte. In the 'pattern' mode, only the top six bits of each byte contain direct intensity information. The two lowest-order intensity bits are replaced by a two-bit pattern which reflects one of four generic subpatterns for the quad. The relevant subpattern assignments are shown in Figure 6-5. The pattern bits are an automatic by-product of the contrast formation step so the compression computation is not significantly complicated by this process.

On decompression, the truncated intensity values are modified by a pattern of bias values which reflect the generic patterns. The bias values are constant over the image but vary among the MAPSel levels. Optimum image-wide biases (in a mean square error sense) can be determined by accumulating simple statistics during compression.

The pattern mode clearly makes no sense for MAPSels at level zero (1x1) since no pattern information is available. Also, the pattern mode is not very effective for large MAPSels since the loss due to intensity truncation from eight to six bits exceeds the size of the optimal biases in most cases. For middle-sized MAPSels, however, the improvement is dramatic. Thus, the user is given the option of selecting which levels will be coded in the 'block' mode and which will use the 'pattern' mode.

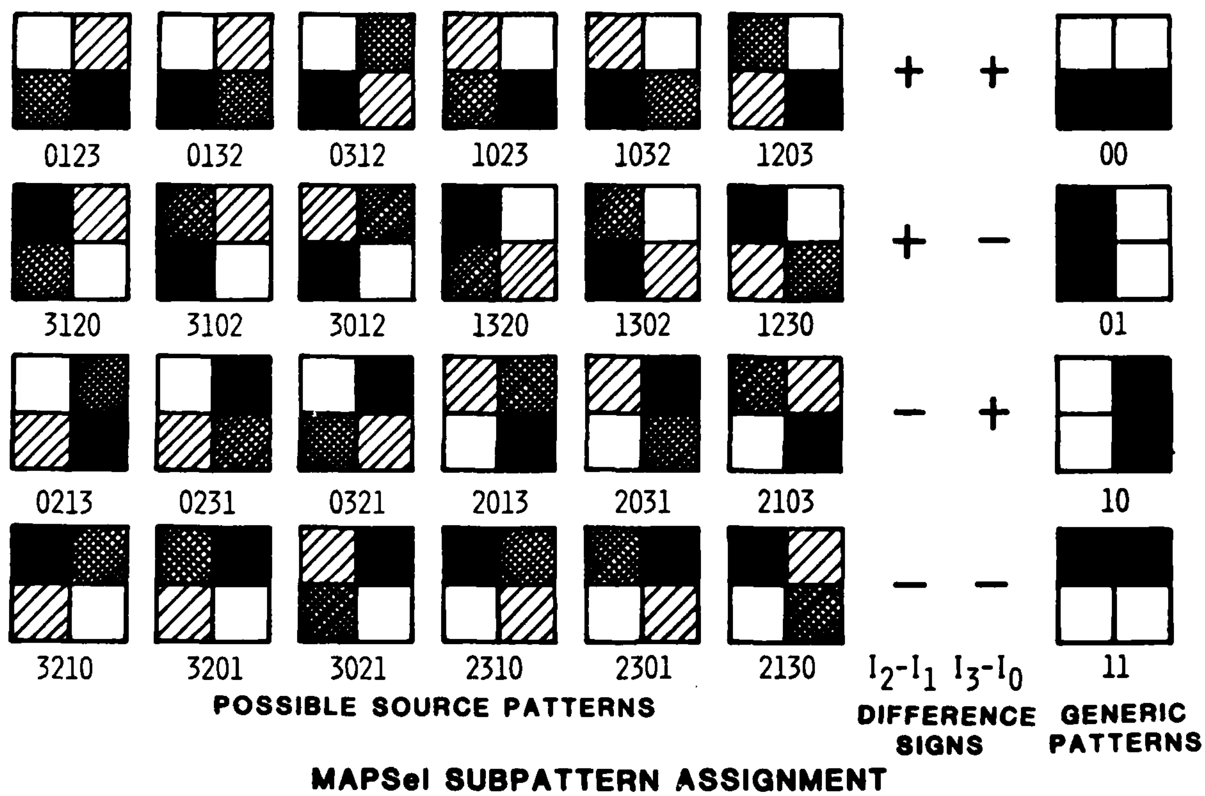


Figure 6-5. MAPS Pattern Mode

6.5 Adaptive 'Convolution'

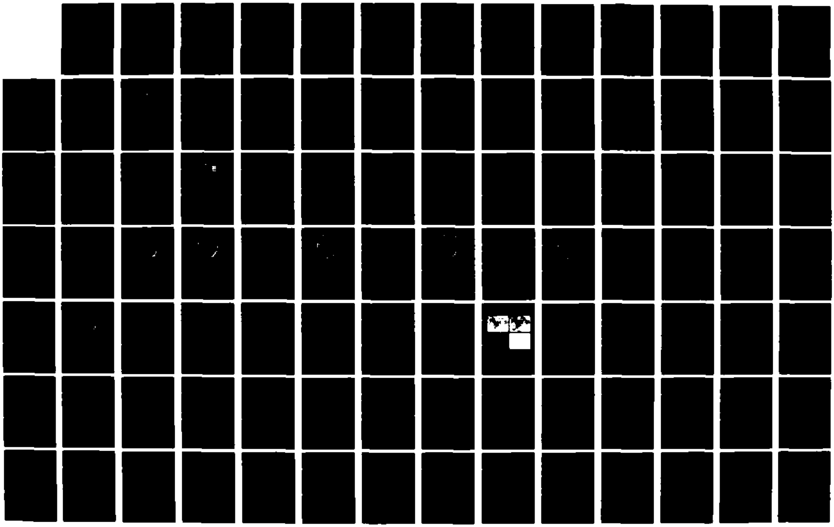
The pattern mode makes dramatic improvements in MAPS image quality. Also, the use of subframe stagger tends to reduce the perception of 'blockiness' in the reconstructed imagery. However, such 'blockiness' can be masked much more effectively through use of a MAPS-based adaptive smoothing process. Note that this adaptive 'convolution' can be used with the pattern mode but is inconsistent with subframe stagger.

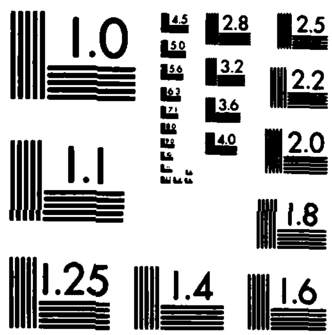
The adaptive smoothing is based on the following general observations. First, the size of a MAPSel is a rough estimate for the local 'correlation' length in the image. Thus, a local 'convolution' window of comparable size is appropriate for image smoothing. Second, the makeup of the 'surround' of the MAPSel to be smoothed contains useful control information for the convolution. Surrounding MAPSels which are much smaller than the target MAPSel give a priori indication of localized image activity which should not be included in the smoothing. Thus, 'surround' elements are activated only if they are no more than one level smaller than the target MAPSel. Finally, this restriction on activation means that the convolution depends on only sixteen regions - four in the (patterned) target MAPSel and twelve in the surround.

The geometry and numbering conventions for the adaptive 'convolution' are shown in Figure 6-6. The dynamic window size is adjusted to be one pixel narrower than the target MAPSel (to make it symmetric about the target pixel). Thus, any target pixel smoothing will depend on at most nine different local and surround elements. Moreover, the window weights can be pre-summed over each of these nine regions for each of the target pixel locations. The convolution then becomes a table-driven process involving just nine regional intensities, nine pre-summed weights, and nine activation flags for each target pixel.

AD-A120 982 TRANSPORTABLE MAPS SOFTWARE VOLUME I(U) CONTROL DATA CORP MINNEAPOLIS MN INFORMATION SCIENCES DIV A E LABONTE JUL 82 9199500 RADC-TR-82-194 2/4

UNCLASSIFIED F30602-80-C-0326 F/G 20/6 NL




1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3

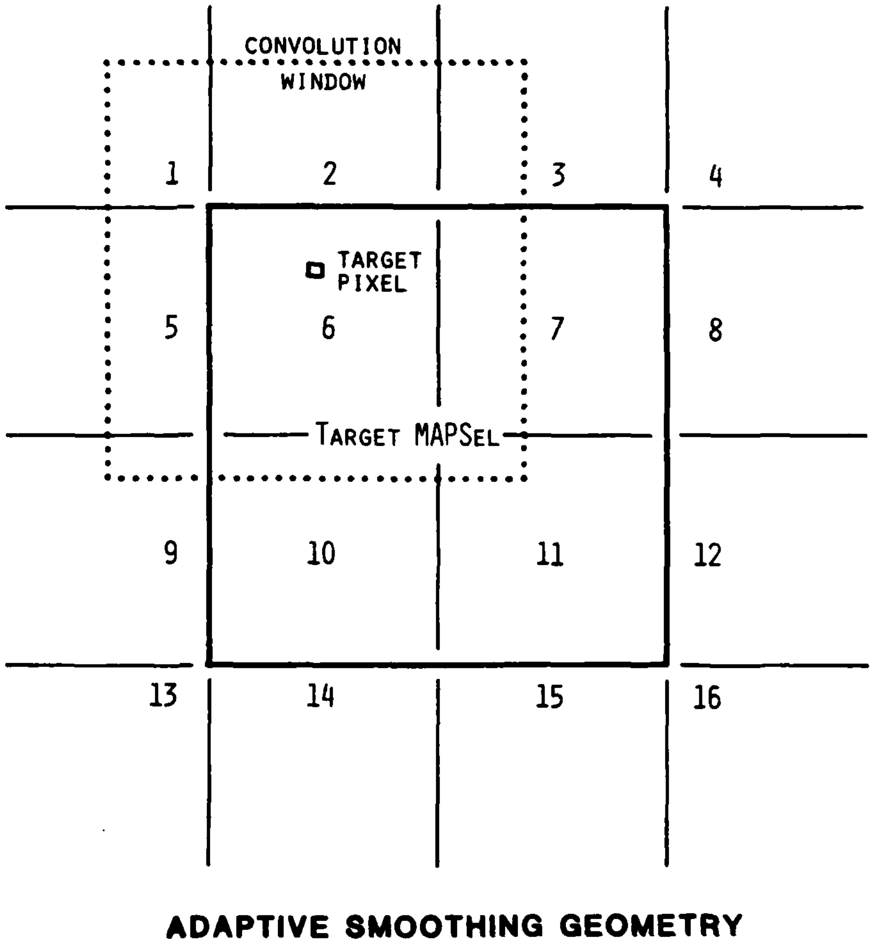


Figure 6-6. MAPS Adaptive 'Convolution'

Since the weights are pre-summed, complex weighting functions can be used at no significant extra expense in computation. The user is given the choice between simple uniform weighting and a two-dimensional Gaussian weight with selectable spread.

SECTION SEVEN

MAPS USER OPTIONS

In this section, all of the user options for modules #1 through #6 are collected and tabulated. Where the role of the option is not clear from the context, a brief discussion is included. The first level of refinement of the total TransMAPS option space is reviewed in the following tabulation:

```
Option Groupings:
------ ---------

        Raster to Subframe Conversion:
        ------ -- -------- ----------
                Source Image Identification
                Source Image Position Specification
                Source Image Size Specification
                Source Image Partition Specification

        MAPS Compression:
        ---- -----------
                User Mode Selection
                Macro-Fidelity Control
                Micro-Fidelity Control
                Gray-Scale Manipulations

        MAPS Decompression and Resolution Image Formation:
        ---- ------------- --- ---------- ----- ---------
                (User Transparent)

        MAPS Adaptive Image Smoothing:
        ---- -------- ----- ---------
                Convolution Weighting Specification
                Dither Amplitude Specification

        MAPS Difference Image Formation:
        ---- ---------- ----- ---------
                Input Image Pair Selection
                Difference Image Control

        Subframe to Raster Conversion:
        -------- -- ------ ----------
                Output Product Image Type Selection

        Image Assembly and Annotation:
        ----- -------- --- ----------
                Output Image Specification
                Embedded Input Image Specifications
                Embedded Annotation Specifications
```

7.1 Raster to Subframe Conversion

The options in program SUBFRM are as follows:

Source Image Identification:
- SOURCE RASTER FILENAME (Up to 9 characters)
- USER IMAGE NAME (Up to 8 characters)

Source Image Position Specification:
- SKIP LINES (Default 0)
- SKIP PIXELS (Default 0)

Source Image Size Specification:
- LINES
- PIXELS (Skip pixels + Retained Pixels $<$ 4001)
- BITS/PIXEL (6 or 8, Default = 8)

Source Image Partition Specification:
- SUBFRAME EDGE (8, 16, or 32, Default = 8)
- SUBFRAME GRID (Square or Stagger, Default = Square)

Constraints: Stagger with 8x8 only,
Stagger incompatible with Adaptive Smoothing.

7.2 MAPS Compression

The MAPS Compression module has a particularly extensive option space. It is subdivided into three option classes - macro-fidelity control, micro-fidelity control, and gray-scale manipulations. The entire interaction is preceded by a 'user mode selection' which determines the overall interaction strategy.

Because of the complexity of the MAPS compression interaction, a separate 'road map' of the decision hierarchy has been included here as Figure 7-1. This presentation shows both the forward penetration of the hierarchy and the structure of return paths when local 'editing' modes are invoked.

7.2.1 User Interaction Mode

Three interaction strategies are available in the MAPS module - the 'Quick' mode, the 'User' mode, and the 'Full' mode. In the 'Quick' mode, only the overall image 'contrast scale' need be selected. This is the normal application of TransMAPS as a 'black box' image coding system.

The 'User' mode allows input of a set of user options defined on a previous interactive session with the MAPS module. The parameter set is stored on file MSET.DAT and overrides the default settings if this mode is selected. Note that the user can establish a new version of MSET.DAT for future use as the last interaction in the current run.

The 'Full' mode allows complete hierarchical penetration of the MAPS option space under user control (see Figure 7-1). Also, a decision to 'Review the Complete Specification' automatically returns the interaction to the beginning of the 'Full' mode sequence. Thus, the user can invoke a previous set-up with the 'User' mode and then edit this further with a 'Review' selection. This is helpful, for example, where a complex macro-fidelity partition is to be held fixed as a various micro-control strategies are explored.

MAPS Compression User Option Decision Hierarchy (Subroutine USERM):

```
Mode Select:  ( Quick  User  Full )

     'Quick' Mode
         Contrast Scale, Group 1

     'User' Mode
         (Input from File MSET.DAT)

     'Full' Mode

         Macro-Fidelity Control:  Review/Revise?
                 (Display Current Image Partition)
     NO          Matrix Row to Change?
                         Revise Selected Row

         Micro-Fidelity Control:  Review/Revise?
             (Loop on Active Groups)
                 (Display Contrast Threshold Parameters, Group k)
                 (Display Contrast Threshold Matrix, Group k)
                 Revise Specifications? (No  Scale-only  Parameters  Matrix)
                     Scale-only:
                         Revise Contrast Scale, Group k
                     Parameters:
                         Revise Contrast Scale, Group k
     NO                  Revise Taper, Group k
                         Revise Step Fraction, Group k
                         Revise Step Bias, Group k
YES                  Matrix:
                         Row to Revise?
                             Revise Matrix Row, Group k
                             (Display Contrast Threshold Matrix, Group k)
             (End Loop)
             (Display Block/Pattern Assignment Vector)
                 Revise Block/Pattern Vector

         Gray-Scale Manipulations:  Review/Revise?
                 Code Space to Contrast Space Remapping:
                         (Display Contrast Remap Breakpoint Pairs)
                         Revise Contrast Remap?
                             (Loop on Breakpoint Pairs)
             NO                  Revise Next Breakpoint Pair
                                 (Exit from Loop if Code Space Exhausted)
                             (End Loop)
     NO          Code Space to Intensity Space Remapping:
                         (Display Intensity Remap Breakpoint Pairs)
                         Revise Intensity Remap?
                             (Loop on Breakpoint Pairs)
             NO                  Revise Next Breakpoint Pair
                                 (Exit from Loop if Code Space Exhausted)
                             (End Loop)
                 (Display Intensity Reset Mode)
                         Revise Intensity Reset: ( M P L S T H )

Review Complete Specification?
 NO
         Retain Specification on File MSET.DAT?
     NO   YES  (Open and Write MSET.DAT)

To MAPS Compression Process
```

Figure 7-1. MAPS Compression Options Overview

7.2.2. Macro-Fidelity Control

Different micro-control strategies can be applied in different parts of the scene by establishing an appropriate macro-fidelity image partition. This involves specifying a 16x16 matrix which divides the source frame into 256 equal-area but distinct rectangular subpatches. The micro-fidelity control for each patch may be selected from any of up to four control parameter groups. Thus, the matrix component corresponding to each patch is assigned one of the digits from 1 to 4.

The normal default is to assign all patches to Group 1. However, the macro-fidelity image partition matrix can be changed on a row by row basis. Each row (16-element vector) update is followed immediately by display of the revised matrix. Thus, the patches can be reassigned until a satisfactory pattern for the scene content has been achieved.

Once the user has signalled an end to the macro-fidelity editing process, the matrix is scanned and all active groups (any subset of the numbers $\{1,2,3,4\}$) are noted. The interaction then automatically queries the user to set just those corresponding micro-fidelity controls.

7.2.3 Micro-Fidelity Control

For each active control group (up to four) there are three modes of control matrix specification available. The user may choose to specify only the 'contrast scale' for the group and leave the other parameters unchanged. Or, the user may elect the 'parametric' mode to change any among the 'contrast scale', 'taper base', 'step fraction', or 'step bias' for the group. A new matrix is then generated and displayed along with the parameter set.

Finally, the user may choose to edit the matrix components directly and by-pass the parametric generation. This approach is particularly appropriate where 'forced compositing' or 'forced resolution retention' is desired. Contrast thresholds which exceed the dynamic range of the possible contrast values will insure forced compositing; threshold values of 255 will work for 8-bit data. On the other hand, negative contrast thresholds will insure forced resolution retention. Examples are given in Sections Eight and Nine below.

The micro-fidelity control interaction also allows selection of the 'block' and 'pattern' mode assignments by MAPSel level. The choice is made by specifying an alphabetic string of B's and P's; the level association proceeds from left to right. Thus, BPPBBB would specify the 'pattern' mode for 2x2 and 4x4 MAPSels with the 'block' mode for the other levels. This particular 'block/pattern' vector is, in fact, the normal default selection.

7.2.4 Gray-Scale Manipulations

The options described in this section have the effect of allowing MAPS control to vary as a function of the image intensity. This is achieved in three ways - by contrast-space remapping, by intensity-space remapping, and by alternative intensity reset strategies.

The effect of a contrast threshold which varies with image intensity can be achieved by keeping a fixed threshold but using a non-linear mapping of the image data from code space (8 bits) to a new contrast space. Furthermore, if the constrast space is coded using more than eight bits, distinctness among the original levels can be preserved even though their relative spacing is changed. This capability is implemented in the MAPS module by allowing the user to specify a piecewise-linear mapping from 8-bit code space to 12-bit contrast space. The mapping can have up to

eight segments, continuous at segment boundaries but with a different slope for each segment.

The user specifies the mapping by responding with successive breakpoint coordinate pairs. That is, a segment end from code space and the corresponding point in contrast space are entered together as a two-element vector. The entry starts at the (0,0) point and proceeds monotonically until the last entered pair exhausts code space (255, < 4096). This approach allows the user to differentially retain certain features, such as radar strong returns, in very high fidelity while smoothing and compressing data from other portions of the gray-scale range.

A similar mapping from 8-bit code space to 12-bit intensity space allows non-linear formation of MAPSel intensities. This is appropriate, for example, if the image 8-bit code space represents a logarithmic encoding of the intensities. In this case, simple averaging of the code values in a quad is equivalent to taking the 'geometric' mean of the original signals rather than the 'arithmetic' mean. This would imply significant and systematic distortion of the radiometric information in the image.

Again, the MAPS module allows a piecewise-linear mapping from code space to intensity space; up to eight segments are also permitted here. This mapping could be used to approximate the transformation from code space back to the original intensity domain. Note that the module automatically forms the inverse demapping from intensity to code space in order to restore the final MAPSels to the proper range.

User specification of the intensity remapping proceeds in exactly the same manner as that for the contrast remapping. In both cases, the user must determine the basis to be used for the remappings before undertaking the interaction.

The final gray-scale manipulation option involves the selection of an intensity reset strategy. Here, the user is given six choices for the manner in which the quad is to be composited from its components. These choices are as follows:

- Mean of the four quad components;
- Pseudo-median of the quad (mean of the two middle elements):
- Lowest intensity in the quad;
- Second-lowest intensity in the quad;
- Third-lowest intensity in the quad;
- Highest intensity in the quad.

This choice may be used to avoid local drop-outs, isolated saturated points, or noise pulses of either sense. It is expected that this facility will be used only in very special image-dependent circumstances. The default selection is the simple mean which is the choice which minimizes the mean square error.

7.3 MAPS Decompression and Resolution Image Formation

As discussed in Section Five, this module proceeds automatically once it is invoked. However, two MAPS products are generated which may require some interpretation. The first is the DMAPS.DAT file which is the MAPS decompressed tonal image in subframe form. The only special feature here is that the optimal pattern biases are automatically applied to all levels for which the 'pattern' mode was selected. The bias information is transmitted as part of the standard MAPS file header.

The second product is the MAPS 'resolution' or 'level' image in file LEVEL.DAT. This is an 'image' formed by placing the host MAPSel resolution code in the upper three bits of each pixel. For 'pattern' mode MAPSels, the two pattern code bits are also included, shifted in two bits from the right edge of the byte. This product, then, gives a visual display of the MAPS resolution coding and is useful in understanding both

the MAPS process and the structure of a particular scene. Examples of the MAPS level image are given in Sections Eight and Nine.

7.4 MAPS Adaptive Smoothing

Two option selections are required in the ADAPT module. They are:

Convolution Weighting:
- UNIFORM or GAUSSIAN (Default Gaussian)
  - SIGMA AT WINDOW CORNER (Gaussian only, Default 2.0)

Dither Selection:
- DITHER AMPLITUDE (Default 4.0)

A small dither may be added in the adaptive smoothing process to mask any residual contouring. The amplitude is in gray levels relative to the eight-bit code scale. The random variable is drawn from the system's pseudo-random number generator.

7.5 MAPS Difference Image Formation

Three option selections are required in the DIFFER module. They are:

First Image of Difference Pair:
- IMAGE.DAT (source) or DMAPS.DAT (MAPS decompressed)

Second Image of Difference Pair:
- DMAPS.DAT or ADAPT.DAT (MAPS smoothed)

Difference Image Control:
- AMPLIFICATION FACTOR (Default 10.0)

If DMAPS.DAT is chosen as the first image, the second automatically defaults to ADAPT.DAT. Thus, any of the three pairs - IMAGE-DMAPS, IMAGE-ADAPT, or DMAPS-ADAPT - can be formed.

The value of the amplification factor controls the type of difference image formed. A negative value results in a 'signed' difference with a neutral gray bias at gray value 127. A positive value results in an 'absolute' difference, amplified by the selected factor. A zero value results in the production of the fidelity statistics only, with no image file formed. The fidelity statistics are output to the printer listing file, EPRINT.DAT.

7.6 Subframe to Raster Conversion

The only option selection required in the RASTER module is selection of the desired product type. The range of possibilities is:

IMAGE.DAT	to	IRAST.DAT	Source image
DMAPS.DAT	to	DRAST.DAT	MAPS decompression
LEVEL.DAT	to	LRAST.DAT	MAPS resolution
ADAPT.DAT	to	ARAST.DAT	Adaptively smoothed
ERROR.DAT	to	ERAST.DAT	MAPS difference

The remaining two sections provide several examples of TransMAPS interactive protocols and sample results.

SECTION EIGHT

MAPS COMPRESSION: EXTENDED USE

This section presents examples of TransMAPS application with emphasis on MAPS compression options.

8.1 MAPS Compression Planning Form

Figure 8-1 presents a reduced photocopy of the MAPS Planning Form for the compression-phase tasks. The planning form summarizes the user option space for modules #1 (SUBFRM) and #2 (MAPS).

Note that the relevant file set for each module is shown. In addition, the three major option subgroups for the MAPS module - macro-fidelity control, micro-fidelity control, and gray-scale manipulations - are clearly distinguished. Finally, formal space is provided at the bottom of the form to record the output summary for the resultant MAPS run.

8.2 TransMAPS Compression Diagnostic Test Image

A special 'toy'-sized diagnostic image was developed as part of TransMAPS to test several aspects of the MAPS compression logic. This image is displayed in Figure 8-2 with an overlay of grid lines to show 'natural' MAPS boundaries relative to the various patterns. Four similar patterns are seen which differ only in scale. These structures contain generic quad geometries for MAPSels of sizes 1x1, 2x2, 4x4, and 8x8. In addition, one quarter of the structure for 16x16 MAPSels and single MAPSels of size 32x32 are represented.

For each of the four smallest MAPSel sizes, all possible generic quad patterns of the following types are included in the twelve quads of the

MAPS PLANNING FORM (COMPRESSION) (ALLOWED RANGE) [DEFAULT] DATE ________________

MODULE — USER INPUTS

RUN SUBFRM

FILES:
- IN - USER RASTER
- OUT - IMAGE.DAT

SOURCE IMAGE FILE NAME (_ _ _ _ _ _ _ _ _ _)
USER IMAGE NAME (_ _ _ _ _ _ _ _)
SKIP LINES [0]________ SKIP PIXELS [0]________
PROCESS LINES[ ]________ PROCESS PIXELS[ ]________
BITS/PIXEL (6 8) [8]
SUBFRAME SIZE (8 16 32) [8] SUBFRAME GRID (SQUARE STAGGER) [SQUARE]
8X8 ONLY

RUN MAPS

FILES:
- IN - IMAGE.DAT
- OUT - MAPS.DAT
- IN/OUT - MSET.DAT
  USER PARAMETERS

MODE (QUICK USER FULL) [Q]

MACRO FIDELITY CONTROL:
IMAGE PARTITION:

PIXEL INDEX DIRECTION →
LINE INDEX DIRECTION ↓
EACH CELL (1 2 3 4) [ALL 1]

MICRO FIDELITY CONTROL:

PARAMETERS	GROUP 1	GROUP 2	GROUP 3	GROUP 4
CONTRAST SCALE	____	____	____	____
TAPER	[3.0]	[3.0]	[3.0]	[3.0]
STEP FRACTION	[0.5]	[0.5]	[0.5]	[0.5]
STEP BIAS	[0.1]	[0.1]	[0.1]	[0.1]

OR:

MATRIX	E M L U	E M L U	E M L U	E M L U
0-1				
1-2				
2-3				
3-4				
4-5				

BLOCK/PATTERN ASSIGNMENT, EACH LEVEL (B P) [BPPBBB]
012345 LEVEL

GRAY-SCALE MANIPULATIONS:

CONTRAST REMAP: CODE CONTRAST
0 0
PIECEWISE LINEAR BREAKPOINT PAIRS [255] [255]
(NON-DECREASING)
CODE (0-255)
CONTRAST (0-4095)
255

INTENSITY REMAP: CODE INTENSITY
0 0
[255] [255]
CODE (0-255)
INTENSITY (0-4095)
255

INTENSITY RESET:
(MEAN)
(PSEUDO-MEDIAN)
(LOWEST)
(SECOND)
(THIRD)
(HIGHEST)
[MEAN]

RESULTS: COMPRESSION: ________
MEAN SQUARE ERROR: ________%

LEVEL:	0	1	2	3	4	5
MAPSELS:						
OPTIMAL BIASES: -						
+						

Figure 8-1. MAPS Compression Planning Form

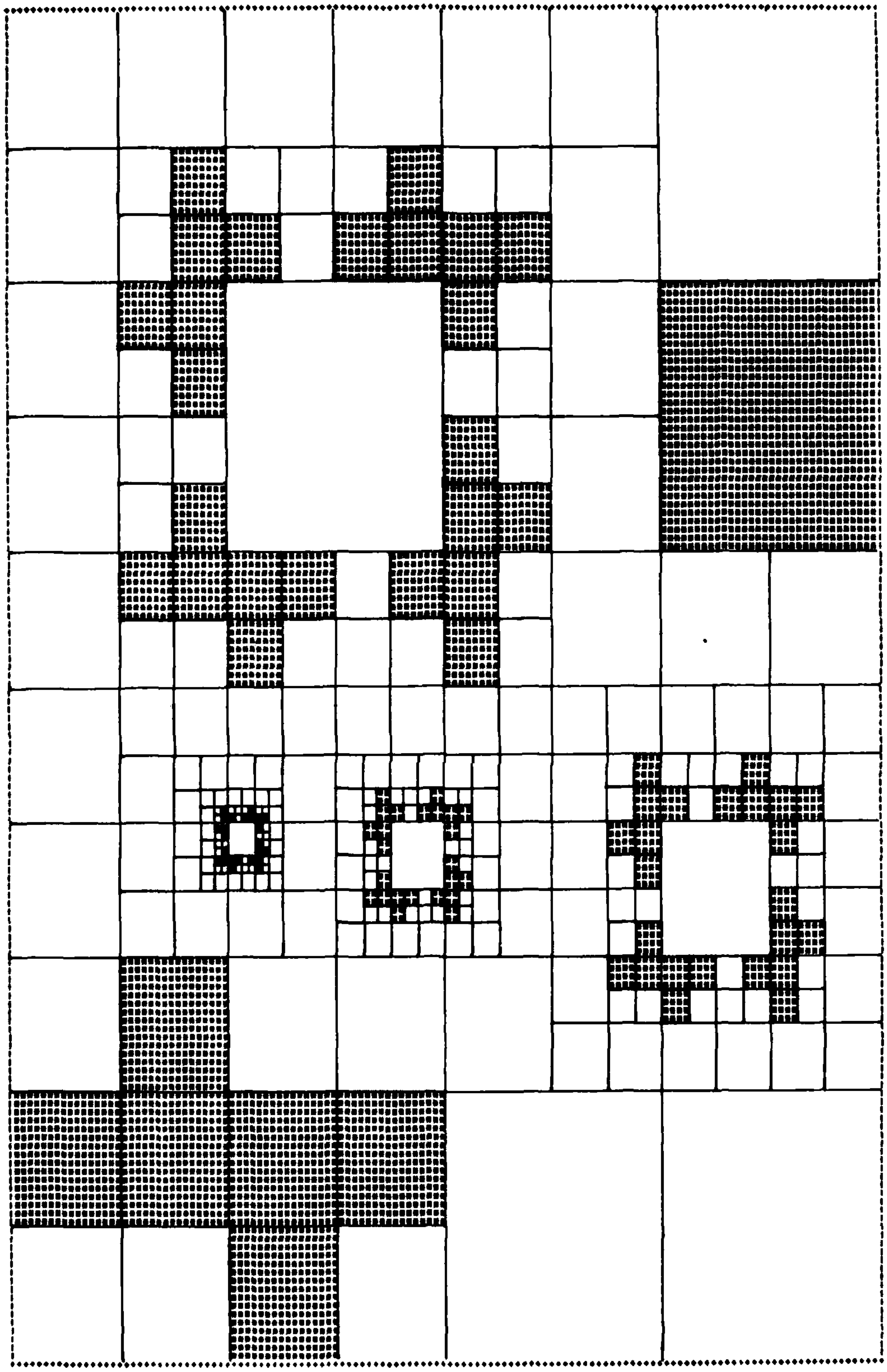


Figure 8-2. MAPS Compression-Logic Diagnostic Image

basic structure:

- one dark MAPSel and three light MAPSels,
- two adjacent dark MAPSels and two adjacent light MAPSels, and
- three dark MAPSels and one light MAPSel.

At least one pattern of each of these types is also included for the 16x16 MAPSel size. Finally, the full frame contains examples of quads with four light MAPSels and others with four dark MAPSels for all five sizes - 1x1, 2x2, 4x4, 8x8, and 16x16.

This diagnostic image can then be used to implement all of the following compression-logic tests:

```
Compression-Logic Diagnostic Tests:
---------------- ---------- ------

        Image Partition and Macro-Fidelity Control:
        ----- --------- --- -------------- --------
                Input Image Line and Pixel Skips
                Subframe Phasing (Square and Staggered)
                Macro-Partition Group Assignment

        Micro-Fidelity Control:
        -------------- --------
                Zigzag Sequencing
                Contrast Control as a Function of Transition Level
                Contrast Control as a Function of Contrast Type
                Pattern Code Assignment

        Gray-Scale Manipulation:
        ---------- ------------
                Contrast Space Quad Sort
                Intensity Space Quad Sort
                Intensity Reset Assignment
```

The diagnostic image is sufficiently small and regular so that the effects of input line and pixel skips can be predicted and verified directly. In addition, the effects of subframe phasing relative to the image structures and the natural MAPS boundaries can be varied by controlled line and pixel skips. These effects can also be predicted for both square and staggered subframes and can be verified very simply by displaying the MAPS resolution image.

Correct performance of the macro-fidelity image partition and corresponding control group assignment can also be established with this image. In this case, if only the first 128 lines of the 160 line by 128 pixel image are used, the macro-fidelity partition will divide the image into a 16x16 pattern of 8x8 pixel patches. Each of the complete scene structures can then be assigned to a different control group and treated with different micro-control strategies. Again, the results can be simply predicted and verified using the MAPS decompression and resolution image products.

Verification of the zig-zag sequencing is implicit in successful reconstruction of the varying MAPSel sizes following 'perfect fidelity' coding using zero contrast thresholds. Verification of contrast control as a function of transition level and contrast type is also possible using combinations of threshold which should yield varying 'forced composition' and 'forced resolution retention'. Finally, correct 'pattern code' assignments can be verified from the various 'two adjacent dark/two adjacent light' configurations which exhaust the generic 'pattern mode' geometries.

Selective combinations of thresholds to give controlled 'forced composition' plus 'forced resolution retention' can also be used to verify the quad sort results in both contrast and intensity space. The nearly exhaustive generic quad configurations also provide the vehicle for verifying the various intensity reset assignments.

Examples of several of these compression option explorations are contained in the protocol presented in the next subsection.

8.3 User Interaction Protocol ('User' and 'Full' Modes)

A completed MAPS-compression planning form for the example in this section is presented in Figure 8-3. The SUBFRM portion of the form

MAPS PLANNING FORM (COMPRESSION) (ALLOWED RANGE) [DEFAULT] DATE *March 1982*

MODULE — **USER INPUTS**

RUN SUBFRM ✓

FILES:
- IN - USER RASTER
- OUT - IMAGE.DAT

SOURCE IMAGE FILE NAME (*MTEST*____)
USER IMAGE NAME (*MTEST_16*)
SKIP LINES [0] *0* SKIP PIXELS [0] *0*
PROCESS LINES [480] *128* PROCESS PIXELS [624] *128*
BITS/PIXEL (6 8) [8]
SUBFRAME SIZE (8 16 32) [8] SUBFRAME GRID (SQUARE STAGGER) [SQUARE]
8X8 ONLY

RUN MAPS ✓

FILES:
- IN - IMAGE.DAT
- OUT - MAPS.DAT
- IN/OUT - MSET.DAT USER PARAMETERS

MODE (QUICK USER FULL) [0]

MACRO FIDELITY CONTROL:
IMAGE PARTITION:
PIXEL INDEX DIRECTION →
LINE INDEX DIRECTION ↓
EACH CELL (1 2 3 4) [ALL 1]

4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
4															4
4															4
4															4
4															4
4															4
4															4
4															4
4															4
4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
4	4	1	1	1	1	2	2	2	2	3	3	3	3	3	3
4	4	1			1	2			2	3					3
4	4	1			1	2			2	3					3
4	4	1	1	1	1	2	2	2	2	3					3
4	4	4	4	4	4	4	4	4	4	3					3
4	4	4	4	4	4	4	4	4	4	3	3	3	3	3	3

MICRO FIDELITY CONTROL:

PARAMETERS	GROUP 1	GROUP 2	GROUP 3	GROUP 4
CONTRAST SCALE				
TAPER	[3.0]	[3.0]	[3.0]	[3.0]
STEP FRACTION	[0.5]	[0.5]	[0.5]	[0.5]
STEP BIAS	[0.1]	[0.1]	[0.1]	[0.1]

OR:

MATRIX	E	M	L	U	E	M	L	U	E	M	L	U	E	M	L	U
0-1	255	0	0	255	0	0	0	0	0	0	0	0	0	0	0	0
1-2	0	0	0	0	-1	-1	-1	-1	0	0	0	0	0	0	0	0
2-3	0	0	0	0	0	0	0	0	255	255	255	255	0	0	0	0
3-4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4-5																

BLOCK/PATTERN ASSIGNMENT, EACH LEVEL (B P) [BPPBBB]
012345 LEVEL
BBBBB

GRAY-SCALE MANIPULATIONS:

CONTRAST REMAP: CODE CONTRAST
PIECEWISE LINEAR BREAKPOINT PAIRS (NON-DECREASING)
CODE (0-255)
CONTRAST (0-4095)
0 0
[255] [255]
255

INTENSITY REMAP: CODE INTENSITY
CODE (0-255)
INTENSITY (0-4095)
0 0
[255] [255]
255

INTENSITY RESET:
(MEAN)
(PSEUDO-MEDIAN)
(LOWEST)
(SECOND)
(THIRD)
(HIGHEST)
[MEAN]

RESULTS: COMPRESSION: *0.231936 b/p*
MEAN SQUARE ERROR: *5.983647*

LEVEL:	0	1	2	3	4	5
MAPSELS:	32	280	8	92	36	
OPTIMAL BIASES: -						
+						

Figure 8-3. Planning Form for MTEST Example

reflects the usage of the diagnostic image, MTEST.DAT. It also indicates that only the first 128 lines (of 160) are to be used here. The image is to be partitioned into a square grid of 16x16 pixel subframes.

The MAPS portion of the planning form exhibits an extensive macro-fidelity-control image partition. The 'scene' has been segmented into four regions, each containing one of the full MAPS test structures. Group 1 is assigned to the patches covering the 1x1-MAPSel structure; group 2 to the 2x2-MAPSel structure; group 3 to the 4x4-MAPSel structure; and the remainder of the frame to group 4. This last group contains the entire 8x8 MAPSel test structure as well as isolated 16x16 and 32x32 MAPSels (see Figure 8-2).

This macro-fidelity partition allows different micro-control strategies to be applied to each of the full test structures. Direct specification of the contrast control matrix is used in each case. For group 1 (and the 1x1-MAPSel structure), two thresholds - the 'extreme' and 'upper step' for the level 0-1 transition - are set to the 'forced compositing' value, 255. All other group 1 thresholds are set to the 'perfect fidelity' value, 0. The effect of this specification should be to combine quads of 1x1 MAPSels which contain 'one dark and three light' components, and to leave the other quad types in the 1x1-MAPSel structure unchanged.

The contrast threshold matrix for group 2 (and the 2x2-MAPSel structure) is set to the 'perfect fidelity' condition except for the level 1-2 transition. This second row is set to the 'forced resolution retention' value, -1, for all four thresholds. Actually, any one of the thresholds set negative is sufficient. This means that the region controlled by group 2 will be coded by MAPSels no larger than 2x2, independent of the local scene content.

The contrast control matrix for group 3 (and the 4x4-MAPSel structure) is set with the 'perfect fidelity' condition in rows 1, 2, and 4. Row 3, corresponding to the level 2-3 transition, is set to the 'forced compositing' value, 255, in all four thresholds. The effect of this should be to composite all quads in the 4x4-MAPSel structure (level 2), to 8x8 MAPSels (level 3).

Finally, the remainder of the image, control group 4, is set completely to the 'perfect fidelity' value, 0. Thus, each element in region 4 should grow to its 'natural' MAPSel size.

Note that the 'block' mode is to be selected for all levels in this example.

Both the 'contrast remap' and 'intensity remap' are to be left in their default form - the identity mapping. However, 'pseudo-median' intensity reset is to be selected.

The results from the actual MAPS module run using these settings are entered at the bottom of the form. Photocopy reproductions of the actual interactive protocols from the resultant DEC-writer listings are displayed below.

The protocol for SUBFRM is as follows:

```
 RUN DR1:[50,27]SF

*********************************************
*                                           *
* MAPS RASTER TO SUBFRAME CONVERSION MODULE *
*                                           *
*********************************************

  SOURCE IDENTIFICATION:

    SOURCE RASTER FILENAME? (UP TO 9 CHARACTERS)  FOR002
MTEST

    USER IMAGE NAME? (UP TO 8 CHARACTERS)
MTEST 16
```

```
  SOURCE IMAGE POSITION:

    NUMBER OF LINES TO SKIP?    0     (/ = NO CHNG)
/
    NUMBER OF PIXELS TO SKIP? (< 4000)    0    (/ = NO CHNG)
/

  SOURCE IMAGE SIZE:

    NUMBER OF LINES TO PROCESS?  480  (/ = NO CHNG)
128
    NUMBER OF PIXELS TO PROCESS? (UP TO 4000)  624  (/ = NO CHNG)
128
    NUMBER OF BITS/PIXEL? (6 8)  8    (/ = NO CHNG)
/

  SOURCE IMAGE PARTITION:

    SUBFRAME EDGE? (8 16 32)  8       (/ = NO CHNG)
16


  USER SPECIFICATION COMPLETE:
  ****************************

    REVIEW? (Y OR N)  N
N

CONVERTING IMAGE MTEST 16 TO     64 SUBFRAMES
>
```

The protocol for MAPS is presented next. Actually, the macro-fidelity image partition matrix had been established on a previous run and saved on file MSET.DAT. (This sample file is also provided as part of the TransMAPS tape.) Thus, the 'User' mode was chosen to re-enter this data, and a 'Y' (yes) response to the 'REVIEW ?" query was used to transfer back to the 'Full' mode for further editing. The MAPS protocol follows:

```
 RUN DR1:[50,27]MP

***************************
*                         *
* MAPS COMPRESSION MODULE *
*                         *
***************************

  USER OPTION MODES:

    Q - QUICK MODE (SELECT CONTRAST SCALE ONLY)
    U - USER PRE-DEFINED PARAMETERS FROM FILE MSET.DAT
    F - FULL OPTION REVIEW AND SELECTIVE REVISION

    MODE? (Q  U  F)  Q
U


  USER SPECIFICATION COMPLETE:
  ****************************

    REVIEW? (Y OR N)  N
Y
```

```
  MACRO-FIDELITY CONTROL:  REVIEW/REVISE? (Y OR N)  N
Y

          CURRENT IMAGE PARTITION

      4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4     ROW  1
      4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4     ROW  2
      4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4     ROW  3
      4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4     ROW  4
      4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4     ROW  5
      4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4     ROW  6
      4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4     ROW  7
      4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4     ROW  8
      4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4     ROW  9
      4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4     ROW 10
      4 4 1 1 1 1 2 2 2 2 3 3 3 3 3 3     ROW 11
      4 4 1 1 1 1 2 2 2 2 3 3 3 3 3 3     ROW 12
      4 4 1 1 1 1 2 2 2 2 3 3 3 3 3 3     ROW 13
      4 4 1 1 1 1 2 2 2 2 3 3 3 3 3 3     ROW 14
      4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 3     ROW 15
      4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 3     ROW 16

   ROW TO CHANGE? (1-16)     (/ = NO FURTHER CHNG)
/

  MICRO-FIDELITY CONTROL:  REVIEW/REVISE? (Y OR N)  N
Y

     GROUP 1 CONTRAST THRESHOLD MATRIX
               E     M     L     U
       0-1     0     0     0     0      ROW 1
       1-2     0     0     0     0      ROW 2
       2-3     0     0     0     0      ROW 3
       3-4     0     0     0     0      ROW 4

   SPECIFICATION MODES:
     N - NO CHANGE
     S - SCALE ONLY
     P - PARAMETRIC
     M - MATRIX

   REVISE SPECIFICATIONS? (N S P M)  N
M

   MATRIX ROW TO CHANGE?  (1-4)     (/ = NO FURTHER CHNG)
1
 REVISE GROUP 1/LEVEL 0-1?
   E    M    L    U
   0    0    0    0     (/ = NO CHNG)
255 0 0 255

     GROUP 1 CONTRAST THRESHOLD MATRIX
               E     M     L     U
       0-1   255     0     0   255      ROW 1
       1-2     0     0     0     0      ROW 2
       2-3     0     0     0     0      ROW 3
       3-4     0     0     0     0      ROW 4

   MATRIX ROW TO CHANGE?  (1-4)     (/ = NO FURTHER CHNG)
/
```

```
      GROUP 2 CONTRAST THRESHOLD MATRIX
                E    M    L    U
        0-1     0    0    0    0      ROW 1
        1-2     0    0    0    0      ROW 2
        2-3     0    0    0    0      ROW 3
        3-4     0    0    0    0      ROW 4


    SPECIFICATION MODES:
      N - NO CHANGE
      S - SCALE ONLY
      P - PARAMETRIC
      M - MATRIX

    REVISE SPECIFICATIONS? (N S P M)  N
M

    MATRIX ROW TO CHANGE?  (1-4)     (/ = NO FURTHER CHNG)
2
 REVISE GROUP 2/LEVEL 1-2?
   E    M    L    U
   0    0    0    0      (/ = NO CHNG)
4*-1

      GROUP 2 CONTRAST THRESHOLD MATRIX
                E    M    L    U
        0-1     0    0    0    0      ROW 1
        1-2    -1   -1   -1   -1      ROW 2
        2-3     0    0    0    0      ROW 3
        3-4     0    0    0    0      ROW 4


    MATRIX ROW TO CHANGE?  (1-4)     (/ = NO FURTHER CHNG)
/


      GROUP 3 CONTRAST THRESHOLD MATRIX
                E    M    L    U
        0-1     0    0    0    0      ROW 1
        1-2     0    0    0    0      ROW 2
        2-3     0    0    0    0      ROW 3
        3-4     0    0    0    0      ROW 4

    SPECIFICATION MODES:
      N - NO CHANGE
      S - SCALE ONLY
      P - PARAMETRIC
      M - MATRIX

    REVISE SPECIFICATIONS? (N S P M)  N
M

    MATRIX ROW TO CHANGE?  (1-4)     (/ = NO FURTHER CHNG)
3
 REVISE GROUP 3/LEVEL 2-3?
   E    M    L    U
   0    0    0    0      (/ = NO CHNG)
4*255

      GROUP 3 CONTRAST THRESHOLD MATRIX
                E    M    L    U
        0-1     0    0    0    0      ROW 1
        1-2     0    0    0    0      ROW 2
        2-3   255  255  255  255      ROW 3
        3-4     0    0    0    0      ROW 4


    MATRIX ROW TO CHANGE?  (1-4)     (/ = NO FURTHER CHNG)
/
```

```
GROUP 4 CONTRAST THRESHOLD MATRIX
           E    M    L    U
    0-1    0    0    0    0      ROW 1
    1-2    0    0    0    0      ROW 2
    2-3    0    0    0    0      ROW 3
    3-4    0    0    0    0      ROW 4

  SPECIFICATION MODES:
    N - NO CHANGE
    S - SCALE ONLY
    P - PARAMETRIC
    M - MATRIX

  REVISE SPECIFICATIONS? (N S P M)  N
N

 BLOCK/PATTERN ASSIGNMENT:

     LEVEL  01234
     MODE   BBBBB

  REVISE B/P VECTOR? BBBBB




GRAY-SCALE MANIPULATIONS:  REVIEW/REVISE? (Y OR N)  N
Y

  CONTRAST SPACE REMAPPING: PIECEWISE LINEAR

    (CODE SPACE/CONTRAST SPACE) BREAKPOINT PAIRS

        0          0
      255        255

  REVISE CONTRAST REMAP? (Y OR N)  N
N

  INTENSITY SPACE REMAPPING: PIECEWISE LINEAR

    (CODE SPACE/INTENSITY SPACE) BREAKPOINT PAIRS

        0          0
      255        255

  REVISE INTENSITY REMAP? (Y OR N)  N
N

  INTENSITY RESET:

    M - MEAN OF QUAD
    P - PSEUDO-MEDIAN OF QUAD
    L - LOWEST IN QUAD
    S - SECOND IN QUAD
    T - THIRD IN QUAD
    H - HIGHEST IN QUAD

  REVISE RESET? (M P L S T H) M
P
```

```
USER SPECIFICATION COMPLETE:
****************************

  REVIEW? (Y OR N)  N
N


SAVE THESE PARAMETERS FOR FUTURE USE? (Y OR N) N
N



MAPS COMPRESSING IMAGE MTEST 16,  128 LINES BY  128 PIXELS









MAPS FILE CONTAINS      1 512-BYTE RECORDS PLUS  86 BYTES IN THE LAST

MAPSEL DISTRIBUTION:

  LEVEL:        0          1          2          3          4
  COUNT:         32        280          8         92         36

  OPTIMAL BIAS: -          0          0          0          0
                +          0          0          0          0

COMPRESSION RATIO:  27.398 : 1

BITS/PIXEL: 0.29199

MEAN SQUARE ERROR:  5.90364 %
>
```

Note that the micro-fidelity control matrices had all been set to the 'perfect fidelity' condition as part of the prior MSET.DAT definition. Also, the 'block/pattern' vector had been set to all 'block' mode, BBBBB. The only editing required was that to revise the contrast control matrices for each group and to update the intensity reset strategy.

The results for this example were retrieved and displayed by invoking modules DMAPS, RASTER (twice), and ANNOTE. The protocol for DMAPS is:

```
RUN DR1:[50,27]DM

*********************************************
*                                           *
* MAPS DECOMPRESSION/RESOLUTION IMAGE MODULE *
*                                           *
*********************************************

  NO USER INPUTS REQUIRED

MAPS DECOMPRESSING IMAGE MTEST 16,  128 LINES BY  128 PIXELS
>
```

The RASTER runs converted both DMAPS.DAT and LEVEL.DAT. The RASTER protocols are:

```
RUN DR1:[50,27]RS

********************************************
*                                          *
* MAPS SUBFRAME TO RASTER CONVERSION MODULE *
*                                          *
********************************************


  MAPS PRODUCT IMAGE TYPE:

      I - IMAGE (ORIGINAL SOURCE)
      D - DMAPS (MAPS DECOMPRESSED)
      L - LEVEL (MAPS RESOLUTION CODES)
      A - ADAPT (ADAPTIVELY SMOOTHED)
      E - ERROR (DIFFERENCE)

    TYPE? (I D L A E)  D[MAPS]
D


  USER SPECIFICATION COMPLETE:
  ****************************

    REVIEW? (Y OR N)  N
N

CONVERTING IMAGE MTEST 16, FILE TYPE: DMAPS

  TO  128 LINE BY  128 PIXEL RASTER, FILE TYPE: DRAST
>
```

```
 RUN DR1:[50,27]RS

*****************************************
*                                       *
* MAPS SUBFRAME TO RASTER CONVERSION MODULE *
*                                       *
*****************************************


  MAPS PRODUCT IMAGE TYPE:

      I - IMAGE (ORIGINAL SOURCE)
      D - DMAPS (MAPS DECOMPRESSED)
      L - LEVEL (MAPS RESOLUTION CODES)
      A - ADAPT (ADAPTIVELY SMOOTHED)
      E - ERROR (DIFFERENCE)

    TYPE? (I D L A E)  D[MAPS]
L


  USER SPECIFICATION COMPLETE:
  ***************************

    REVIEW? (Y OR N)  N
N

CONVERTING IMAGE MTEST 16, FILE TYPE: LEVEL

  TO  128 LINE BY  128 PIXEL RASTER, FILE TYPE: LRAST
>
```

Finally, the ANNOTE protocol is:

```
 RUN DR1:[50,27]AI

****************************************
*                                      *
* IMAGE ASSEMBLY AND ANNOTATION MODULE *
*                                      *
****************************************


  OUTPUT IMAGE SPECIFICATION:

    OUTPUT FILE MODE:

      R  -  GRAY SCALE RASTER IMAGE FILE 'ANIMG.DAT'
      P  -  LINE PRINTER PSEUDO IMAGE FILE 'APRINT.DAT'

    MODE?  (R P)  R
P

    NUMBER OF LINES?  800     (/ = NO CHNG)
257
    NUMBER OF PIXELS? (UP TO  128)  128    (/ = NO CHNG)
/
    COMPLEMENT BACKGROUND? (Y OR N)  N
N
```

```
  EMBEDDED IMAGES:
>
    NUMBER OF IMAGES? (0 1 2)  0     (/ = NO CHNG)
2
    IMAGE 1:
      FILENAME? (UP TO 9 CHARACTERS)  FOR002
DRAST
      SKIP LINES INTO INPUT IMAGE?    0     (/ = NO CHNG)
/
      SKIP PIXELS INTO INPUT IMAGE? (<4000)    0     (/ = NO CHNG)
/
      NUMBER OF LINES? (UP TO  257)   257     (/ = NO CHNG)
128
      NUMBER OF PIXELS? (UP TP  128)   128     (/ = NO CHNG)
/
      STARTING LINE? (RANGE 1 - 130)     1     (/ = NO CHNG)
/
      STARTING PIXEL? (RANGE 1 -   1)     1     (/ = NO CHNG)
/
      COMPLEMENT IMAGE? (Y OR N)  N
N
    IMAGE 2:
      FILENAME? (UP TO 9 CHARACTERS)  FOR003
LRAST
      SKIP LINES INTO INPUT IMAGE?    0     (/ = NO CHNG)
/
      SKIP PIXELS INTO INPUT IMAGE? (<4000)    0     (/ = NO CHNG)
/
      NUMBER OF LINES? (UP TO  257)   257     (/ = NO CHNG)
128
      NUMBER OF PIXELS? (UP TP  128)   128     (/ = NO CHNG)
/
      STARTING LINE? (RANGE 1 - 130)     1     (/ = NO CHNG)
?30
      STARTING PIXEL? (RANGE 1 -   1)     1     (/ = NO CHNG)
/
      COMPLEMENT IMAGE? (Y OR N)  N
N

  EMBEDDED ANNOTATION:

    NUMBER OF MESSAGES? (0 -  20)   0     (/ = NO CHNG)
/
  USER SPECIFICATION COMPLETE:
  ****************************

    REVIEW? (Y OR N)  N
N
ASSEMBLING AND ANNOTATING IMAGE:

      257 LINES BY  128 PIXELS TO FILE 'APRINT.DAT'
>
```

The resultant MAPS 'decompressed' image and the corresponding MAPS 'level' image are displayed in Figure 8-4. For the 1x1-MAPSel structure, the 'one dark and three light' quads are seen to be forced to 2x2 form and to show up as 'light' MAPSels. This intensity reset is a consequence of the 'pseudo-median' selection and the fact that the two 'middle' intensities are both 'light'. The other quads of 1x1 MAPSels are left unchanged. Thus, the predictions for this structure are all borne out.

In the region corresponding to group 2, the resolution image shows that 2x2 MAPSels are used throughout. Again, the predictions are verified.

In the group 3 region, all 4x4 MAPSels are seen to be composited to 8x8 form. The 'one dark and three light' quads go to all 'light'. The 'two adjacent dark and two adjacent light' quads go to a 'mid-gray'. The 'three dark and one light' quads go to all 'dark'. Each of these is consistent with 'forced compositing' and 'pseudo-median reset' as expected.

Finally, region 4 shows the predicted 'perfect fidelity' decompression and 'natural MAPSel' resolution-image structure.

This example demonstrates the diagnostic and verification power of the special test image, MTEST.DAT. It also shows the potential of this image to illustrate the detailed performance of the MAPS compression options. The user is encouraged to exploit this image for further development of MAPS process understanding and 'intuition'.

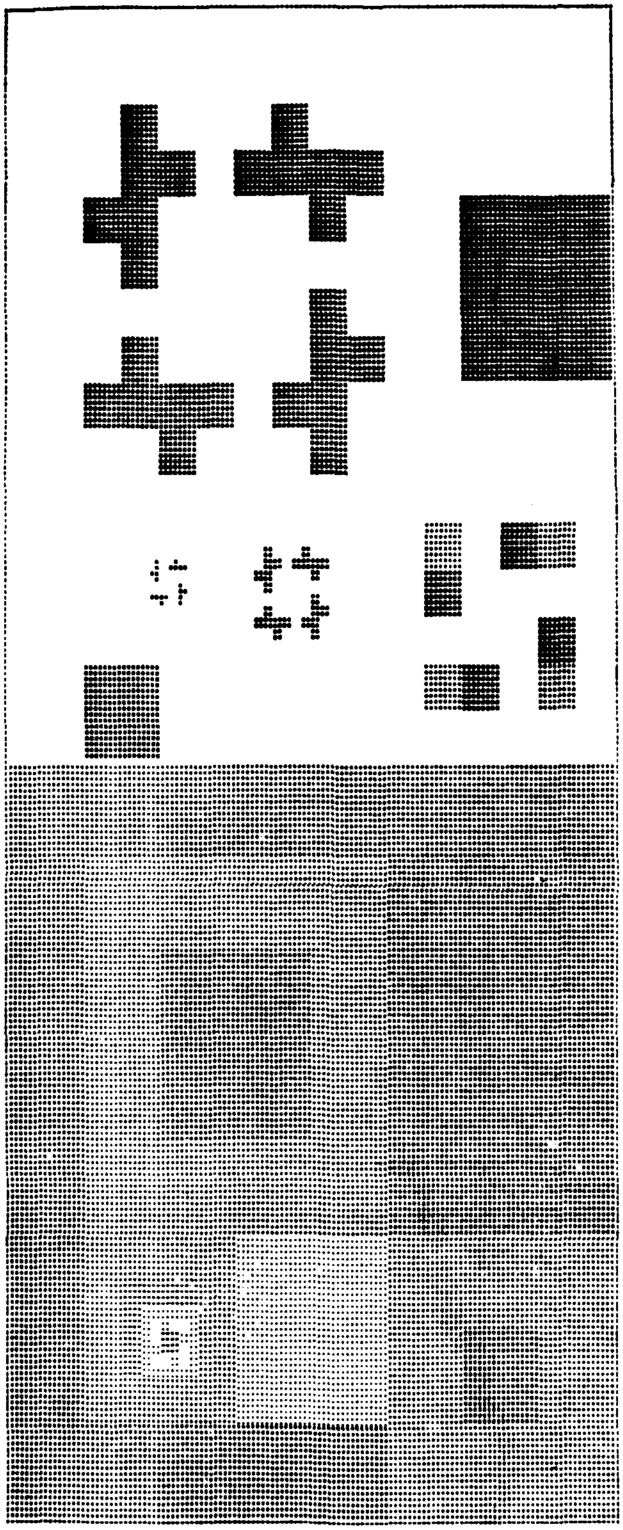


Figure 8-4. MAPS Decompressed and Resolution Images for Diagnostic Test Example

SECTION NINE

MAPS PRODUCT GENERATION: EXTENDED USE

This section presents examples of TransMAPS application with emphasis on MAPS Product formation options.

9.1 MAPS Product Generation Planning Form

Figure 9-1 presents a reduced photocopy of the MAPS Planning Form for the product generation tasks. The planning form summarizes the user options for modules #3 (DMAPS), #4 (ADAPT), #5 (DIFFER) and #6 (RASTER).

These modules have relatively few interactive parameters compared with the compression-phase tasks. However, the file structure is much more extensive and complex (see Figure 2-1). The relevant files for each module are listed on the form.

9.2 User Interaction Protocol (All Modules)

The remaining subsections describe a series of examples in which the MAPS compression level is held fixed at essentially two bits per pixel. The source image for the series is the 'toy' (120 line x 128 pixel) GIRL6.DAT image. The examples depict the evolution of increasing MAPS image 'quality' under alternative compression and product generation strategies. Seven examples make up the series and the comparative control states are summarized in the following tabulation:

MAPS PLANNING FORM (PRODUCT GENERATION) (ALLOWED RANGE) [DEFAULT]

DATE________________

IMAGE NAME__________

COMPRESSION_________

MODULE | USER INPUTS

RUN DMAPS

FILES:

IN - MAPS.DAT

OUT - DMAPS.DAT

OUT - LEVEL.DAT

RUN ADAPT /NOT WITH STAGGER GRID/

FILES:

IN - DMAPS.DAT

IN - LEVEL.DAT

OUT - ADAPT.DAT

CONVOLUTION WEIGHTING (GAUSSIAN UNIFORM) [G]

SIGMA MULTIPLE AT WINDOW CORNER [2.0] ____ /GAUSSIAN ONLY/

RANDOM DITHER AMPLITUDE [4.0] ___

RUN DIFFER

FILES:

IN - IMAGE.DAT
IN - DMAPS.DAT
IN - ADAPT.DAT
} ANY TWO

OUT - ERROR.DAT

OUT - EPRINT.DAT (LISTING)

FILE PAIRINGS (IMAGE vs DMAPS IMAGE vs ADAPT DMAPS vs ADAPT) [1 D]

DIFFERENCE PARAMETER:

(<0 SIGNED =0 STATISTICS ONLY >0 AMPLIFICATION FACTOR)[10]____

RUN RASTER

FILES:

IN - IMAGE.DAT
IN - DMAPS.DAT
IN - LEVEL.DAT
IN - ADAPT.DAT
IN - ERROR.DAT
} ANY ONE

OUT - IRAST.DAT
OUT - DRAST.DAT
OUT - LRAST.DAT
OUT - ARAST.DAT
OUT - ERAST.DAT
} TYPE MATCHES INPUT

MAPS RASTER PRODUCT (IMAGE DMAPS LEVEL ADAPT ERROR) [DMAPS]

Figure 9-1. MAPS Product Generation Planning Form

Product Generation Examples:

Example	Contrast Scale	Block/Pattern	Convolution	Dither
2X2 MEAN	Force Level 1	-	-	-
DMAPS B2	72	BBBB	-	-
ADAPT B2	72	BBBB	Uniform	0
DMAPS P2	72	BPPP	-	-
ADAPT P2	72	BPPP	Uniform	0
GAUSS P2	72	BPPP	Gaussian;2	0
DITHER 8	72	BPPP	Gaussian;2	8

A sample protocol from one of the examples is exhibited as the remainder of this subsection. The protocol corresponds to the 'GAUSS P2' case and involves all seven modules in TransMAPS. The sequence of module invocations is as follows:

```
RUN  SF
RUN  MP
RUN  DM
RUN  AD
RUN  DF
RUN  RS          (Convert ADAPT.DAT)
RUN  RS          (Convert ERROR.DAT)
RUN  AI
```

By this point, the protocols should be self-explanatory so they are displayed sequentially without interruption:

```
 RUN DR1:[50,27]SF
******************************************
*                                        *
* MAPS RASTER TO SUBFRAME CONVERSION MODULE *
*                                        *
******************************************

  SOURCE IDENTIFICATION:

    SOURCE RASTER FILENAME? (UP TO 9 CHARACTERS)  FOR002
GIRL6

    USER IMAGE NAME? (UP TO 8 CHARACTERS)
GIRL 8x8
```

```
  SOURCE IMAGE POSITION:

    NUMBER OF LINES TO SKIP?    0      (/ = NO CHNG)
/
    NUMBER OF PIXELS TO SKIP? (< 4000)    0     (/ = NO CHNG)
/

  SOURCE IMAGE SIZE:

    NUMBER OF LINES TO PROCESS?  480  (/ = NO CHNG)
120
    NUMBER OF PIXELS TO PROCESS? (UP TO 4000)  624  (/ = NO CHNG)
128
    NUMBER OF BITS/PIXEL? (6 8)  8    (/ = NO CHNG)
6

  SOURCE IMAGE PARTITION:

    SUBFRAME EDGE? (8 16 32)  8       (/ = NO CHNG)
/
    STAGGER GRID? (Y OR N) N
N


  USER SPECIFICATION COMPLETE:
  ****************************

    REVIEW? (Y OR N)  N
N

CONVERTING IMAGE GIRL 8x8 TO    240 SUBFRAMES
>




***************************
*                         *
* MAPS COMPRESSION MODULE *
*                         *
***************************

  USER OPTION MODES:

    Q - QUICK MODE (SELECT CONTRAST SCALE ONLY)
    U - USER PRE-DEFINED PARAMETERS FROM FILE MSET.DAT
    F - FULL OPTION REVIEW AND SELECTIVE REVISION

    MODE? (Q  U  F)  Q
F


  MACRO-FIDELITY CONTROL:  REVIEW/REVISE? (Y OR N)  N
N
```

```
  MICRO-FIDELITY CONTROL:  REVIEW/REVISE? (Y OR N)   N
Y
      GROUP 1 CONTRAST THRESHOLD PARAMETERS
        CONTRAST SCALE   20.0
        TAPER             3.0
        STEP FRACTION     0.5
        STEP BIAS         0.1

      GROUP 1 CONTRAST THRESHOLD MATRIX
                E    M    L    U
        0-1    20   10   12   12     ROW 1
        1-2     7    3    4    4     ROW 2
        2-3     2    1    1    1     ROW 3


    SPECIFICATION MODES:
      N - NO CHANGE
      S - SCALE ONLY
      P - PARAMETRIC
      M - MATRIX

    REVISE SPECIFICATIONS? (N S P M)  N
S
    GROUP 1 CONTRAST SCALE?   20.0     (/ = NO CHNG)
72
      GROUP 1 CONTRAST THRESHOLD PARAMETERS
        CONTRAST SCALE   72.0
        TAPER             3.0
        STEP FRACTION     0.5
        STEP BIAS         0.1

      GROUP 1 CONTRAST THRESHOLD MATRIX
                E    M    L    U
        0-1    72   36   43   43     ROW 1
        1-2    24   12   14   14     ROW 2
        2-3     8    4    5    5     ROW 3


    SPECIFICATION MODES:
      N - NO CHANGE
      S - SCALE ONLY
      P - PARAMETRIC
      M - MATRIX

    REVISE SPECIFICATIONS? (N S P M)  N
N

  BLOCK/PATTERN ASSIGNMENT:

        LEVEL  0123
        MODE   BPPB

    REVISE B/P VECTOR? BPPB
BPPP
```

```
  GRAY-SCALE MANIPULATIONS:  REVIEW/REVISE? (Y OR N)  N
N

  USER SPECIFICATION COMPLETE:
  ****************************

    REVIEW? (Y OR N)  N
N
  SAVE THESE PARAMETERS FOR FUTURE USE? (Y OR N) N
N


MAPS COMPRESSING IMAGE GIRL 8x8,  120 LINES BY  128 PIXELS


MAPS FILE CONTAINS     7 512-BYTE RECORDS PLUS 234 BYTES IN THE LAST

MAPSEL DISTRIBUTION:

  LEVEL:       0        1        2        3
  COUNT:     1052     1557      425       20

  OPTIMAL BIAS: -      -6       -2        0
                +       9        5        3

COMPRESSION RATIO:   3.017 : 1

BITS/PIXEL: 1.98854

MEAN SQUARE ERROR:  0.18080 %
>




 RUN DR1:[50,27]DM

********************************************
*                                          *
* MAPS DECOMPRESSION/RESOLUTION IMAGE MODULE *
*                                          *
********************************************

  NO USER INPUTS REQUIRED

MAPS DECOMPRESSING IMAGE GIRL 8x8,  120 LINES BY  128 PIXELS
>
```

```
 RUN DR1:[50,27]AD
*********************************
*                               *
* MAPS ADAPTIVE SMOOTHING MODULE *
*                               *
*********************************


  CONVOLUTION WEIGHTING:

    UNIFORM OR GAUSSIAN? (U G)  G
G
    SIGMA MULTIPLE AT WINDOW CORNER?  2.0  (/ = NO CHNG)
/

  RANDOM DITHER:

    AMPLITUDE?  4.0                        (/ = NO CHNG)
0


  USER SPECIFICATION COMPLETE:
  ****************************

    REVIEW? (Y OR N)  N
N

MAPS ADAPTIVE SMOOTHING IMAGE GIRL 8x8,  120 LINES BY  128 PIXELS
>


RUN DR1:[50,27]DF

********************************
*                              *
* MAPS DIFFERENCE IMAGE MODULE *
*                              *
********************************


  INPUT IMAGE TYPES:

     I - IMAGE (ORIGINAL SOURCE) IMAGE1 ONLY
     D - DMAPS (MAPS DECOMPRESSED)
     A - ADAPT (ADAPTIVELY SMOOTHED) IMAGE2 ONLY

    IMAGE1? (I D)  I[ IAGE]
I
    IMAGE2? (D A)  D[MAPS]
A

  DIFFERENCE IMAGE TYPE:

     AMPLIFICATION FACTOR
       <0  SIGNED AND BIASED DIFFERENCE IMAGE
       =0  NO DIFFERENCE IMAGE, STATISTICS ONLY
       >0  AMPLIFIED DIFFERENCE IMAGE
              (VALUE IS AMPLIFICATION)

    FACTOR?  10.     (/ = NO CHNG)
6


  USER SPECIFICATION COMPLETE:
  ****************************

    REVIEW? (Y OR N)  N
N

DIFFERENCING GIRL 8x8 TYPE IMAGE, VS GIRL 8x8 TYPE ADAPT
>
```

```
 RUN DR1:[50,27]RS

*********************************************
*                                           *
* MAPS SUBFRAME TO RASTER CONVERSION MODULE *
*                                           *
*********************************************


  MAPS PRODUCT IMAGE TYPE:

      I - IMAGE (ORIGINAL SOURCE)
      D - DMAPS (MAPS DECOMPRESSED)
      L - LEVEL (MAPS RESOLUTION CODES)
      A - ADAPT (ADAPTIVELY SMOOTHED)
      E - ERROR (DIFFERENCE)

    TYPE? (I D L A E)  D[MAPS]
A


  USER SPECIFICATION COMPLETE:
  ****************************

    REVIEW? (Y OR N)  N
N

CONVERTING IMAGE GIRL 8x8, FILE TYPE: ADAPT

  TO  120 LINE BY  128 PIXEL RASTER, FILE TYPE: ARAST
>

 RUN DR1:[50,27]RS

*********************************************
*                                           *
* MAPS SUBFRAME TO RASTER CONVERSION MODULE *
*                                           *
*********************************************


  MAPS PRODUCT IMAGE TYPE:

      I - IMAGE (ORIGINAL SOURCE)
      D - DMAPS (MAPS DECOMPRESSED)
      L - LEVEL (MAPS RESOLUTION CODES)
      A - ADAPT (ADAPTIVELY SMOOTHED)
      E - ERROR (DIFFERENCE)

    TYPE? (I D L A E)  D[MAPS]
E


  USER SPECIFICATION COMPLETE:
  ****************************

    REVIEW? (Y OR N)  N
N

CONVERTING IMAGE GIRL 8x8, FILE TYPE: ERROR A-I

  TO  120 LINE BY  128 PIXEL RASTER, FILE TYPE: ERAST
>
```

```
 RUN DR1:[50,27]AI

****************************************
*                                      *
* IMAGE ASSEMBLY AND ANNOTATION MODULE *
*                                      *
****************************************

  OUTPUT IMAGE SPECIFICATION:

    OUTPUT FILE MODE:

       R  -  GRAY SCALE RASTER IMAGE FILE 'ANIMG.DAT'
       P  -  LINE PRINTER PSEUDO IMAGE FILE 'APRINT.DAT'

    MODE?  (R P)  R
P

    NUMBER OF LINES?  800     (/ = NO CHNG)
256
    NUMBER OF PIXELS? (UP TO  128)  128    (/ = NO CHNG)
/
    COMPLEMENT BACKGROUND? (Y OR N)  N


  EMBEDDED IMAGES:

    NUMBER OF IMAGES? (0 1 2)  0     (/ = NO CHNG)
2
    IMAGE 1:

      FILENAME? (UP TO 9 CHARACTERS)  FOR002
ARAST
      SKIP LINES INTO INPUT IMAGE?    0     (/ = NO CHNG)
/
      SKIP PIXELS INTO INPUT IMAGE? (<4000)    0     (/ = NO CHNG)
/
      NUMBER OF LINES? (UP TO  256)   256     (/ = NO CHNG)
120
      NUMBER OF PIXELS? (UP TP  128)   128     (/ = NO CHNG)
/
      STARTING LINE? (RANGE 1 - 137)     1     (/ = NO CHNG)
/
      STARTING PIXEL? (RANGE 1 -   1)     1     (/ = NO CHNG)
/
      COMPLEMENT IMAGE? (Y OR N)  N
N
    IMAGE 2:

      FILENAME? (UP TO 9 CHARACTERS)  FOR003
ERAST
      SKIP LINES INTO INPUT IMAGE?    0     (/ = NO CHNG)
/
      SKIP PIXELS INTO INPUT IMAGE? (<4000)    0     (/ = NO CHNG)
/
      NUMBER OF LINES? (UP TO  256)   256     (/ = NO CHNG)
120
      NUMBER OF PIXELS? (UP TP  128)   128     (/ = NO CHNG)
/
      STARTING LINE? (RANGE 1 - 137)     1     (/ = NO CHNG)
137
      STARTING PIXEL? (RANGE 1 -   1)     1     (/ = NO CHNG)
/
      COMPLEMENT IMAGE? (Y OR N)  N
N
```

```
  EMBEDDED ANNOTATION:

    NUMBER OF MESSAGES? (0 -  20)   0     (/ = NO CHNG)
1

    MESSAGE  1:

      ORIENTATION IN FRAME, TOP OF SYMBOL TOWARD:
                                T - TOP
                                B - BOTTOM
                                L - LEFT
                                R - RIGHT

        ORIENTATION? (T B L R)  T
T
      MESSAGE  1 LENGTH
          0 -  8 CHARACTERS AT 1X
          0 -  4 CHARACTERS AT 2X
          0 -  2 CHARACTERS AT 3X
          0 -  2 CHARACTERS AT 4X

        CHARACTER COUNT?   0     (/ = NO CHANGE, 0 = DELETE)
8
      MESSAGE  1 SYMBOL SIZE? (1      )  1     (/ = NO CHNG)
/
      MESSAGE CENTER AT LINE? (   8 -  248)    8     (/ = NO CHNG)
128
      MESSAGE CENTER AT PIXEL? (  64 -   64)   64     (/ = NO CHNG)
/
      COMPLEMENT MESSAGE  1? (Y OR N)  N
N
    ALLOWED CHARACTERS

      ALPHA CAPS:  A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
      NUMERALS:    0 1 2 3 4 5 6 7 8 9
      PUNCTUATION: ( ) . , : Space
      ARITHMETIC:  + - * / %
      RELATIONAL:  = < > f(<or=) s(>or=)
      DIRECTIONAL: e (East)
        (ARROWS)   n (North)                n
                   w (West)             b       a
                   s (South)
                   a (NE Northeast)   w     +     e
                   b (NW Northwest)
                   c (SW Southwest)     c       d
                   d (SE Southeast)         s

      MESSAGE  1 TEXT? ( 8 CHARACTERS)


--------
GAUSS P2


MSG    LINE  PIX    TEXT
 1 T 1X  128   64   GAUSS P2

  MESSAGE TO CHANGE? (1 - 1)     (/ = NO FURTHER CHNG)
/

  USER SPECIFICATION COMPLETE:
  ****************************

    REVIEW? (Y OR N)  N
N

ASSEMBLING AND ANNOTATING IMAGE:

      256 LINES BY  128 PIXELS TO FILE 'APRINT.DAT'
>
```

9.3 Resolution (level) Image

Figure 9-2 shows the source frame and the resolution or level image which exhibits the same form for all of the runs at contrast scale 72. The white areas correspond to 1x1 coding, the 'dots' to 2x2's, the 'minuses' to 4x4's and the 'pluses' to 8x8's. Note that MAPS concentrates its resources at points which define the key features of the image content.

Figure 9-3 presents the results of simple 2x2 averaging to achieve two bits per pixel. This then forms the baseline for comparison with MAPS products at the same compression level. This case was established using TransMAPS with 'forced compositing' (threshold 255) for the level 0-1 transition and 'forced resolution retention' (threshold -1) for the level 1-2 transition.

The lower half of Figure 9-3 portrays the results of the DIFFER Module applied between the source (1x1 throughout) and the 2x2 MEAN image. An amplification factor of 6.0 was used. The corresponding error histogram is displayed in Figure 9-4; this is the EPRINT.DAT file from DIFFER.

9.4 Block Decompression

Figure 9-5 shows the MAPS decompression and amplified difference image (factor = 6.0) for simple 'block' mode compression at all levels. The error histogram is shown in Figure 9-6. Note that MAPS definition around the eyes, mouth, and jaw line is considerably sharper than in the 2x2 MEAN case. However, the image is quite 'blocky' elsewhere.

9.5 Block Mode with Uniform Adaptive Smoothing

Figure 9-7 corresponds to subsequent adaptive smoothing of the 'block' mode results in Figure 9-5. The comparable error histogram is given in Figure 9-8. Note that the smoothing process does much to eliminate the perceptible artifacts. However, the smoothing process is relatively expensive in computation time.

9.6 Pattern Decompression

Figure 9-9 shows the MAPS decompression and difference image with the 'pattern' mode used throughout. The error histogram is given in Figure 9-10. This case shows significant sharpening of features relative to the 'block' mode and is quite acceptable even without further smoothing. The difference image exhibits a mostly 'salt and pepper' texture which is largely decorrelated from the scene content. Note that this is the normal default mode recommended for MAPS use and is very efficient computationally.

9.7 Pattern Mode with Uniform Adaptive Smoothing

Figure 9-11 corresponds to the adaptive smoothing of the 'pattern' image in Figure 9-9; the error histograms are shown in Figure 9-12. Here, 'uniform' convolution weighting was used in the smoothing process. This version is slightly better than the direct 'pattern' mode but at considerable extra computation.

9.8 Pattern Mode with Gaussian Adaptive Smoothing

Figure 9-13 depicts the same case as Figure 9-11 except that 'Gaussian' convolution weighting was substituted for 'uniform' weights. The spread of the weighting function was chosen so that the 2.0 sigma point occurs in the corner of the convolution window. Very small differences can be discerned between the results in Figures 9-11 and 9-13. The error histogram for the Gaussian case is presented in Figure 9-14.

9.9 Pattern/Gaussian Mode with Dither

Figure 9-15 repeats the Gaussian-smoothed results except that a random dither has been added to the 'smoothing' process. The maximum amplitude of the dither is 8 gray levels (of 256) with the actual sign and size for each pixel determined from a pseudo-random number generator. Figure 9-16 gives the matching error histogram. This example is seen to have less perceptible intensity 'contouring' than the cases without dither.

Figure 9-2. MAPS Resolution Image at 2 bits/pixel

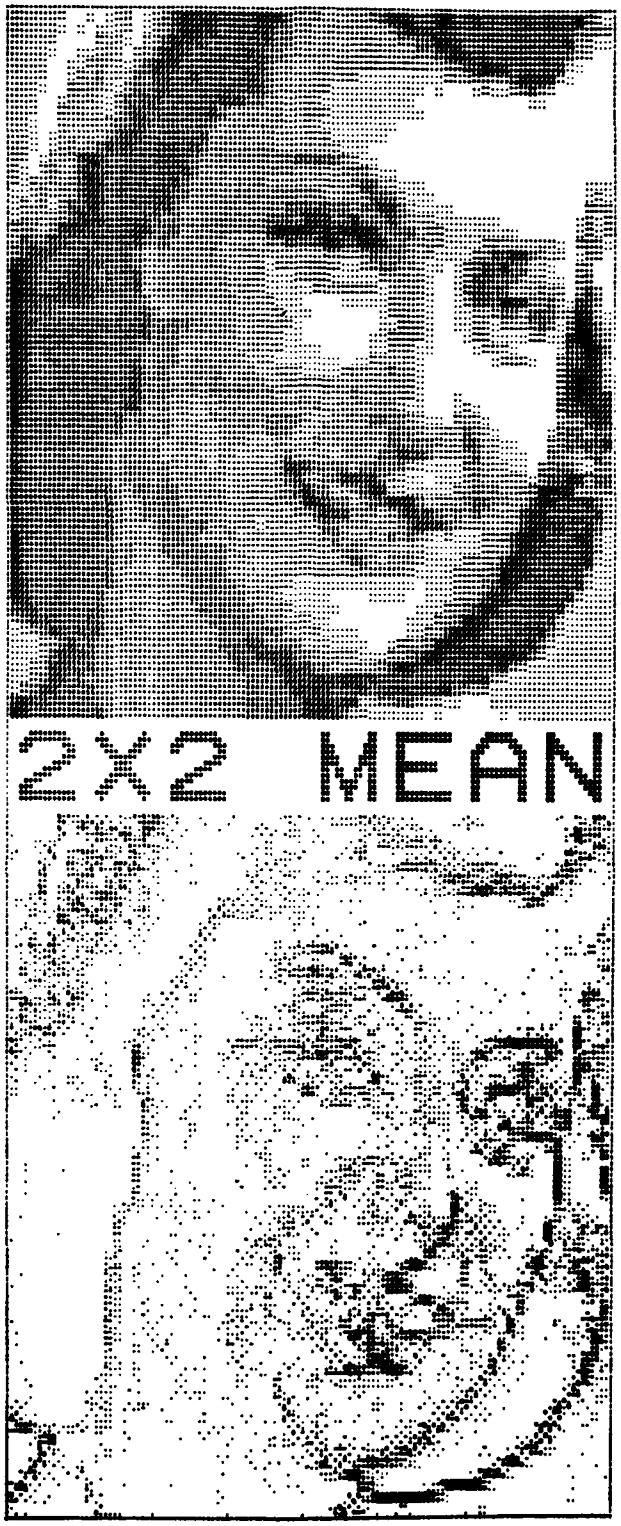


Figure 9-3. Simple Averaging to Achieve 2 bits/pixel

DIFFERENCE STATISTICS:

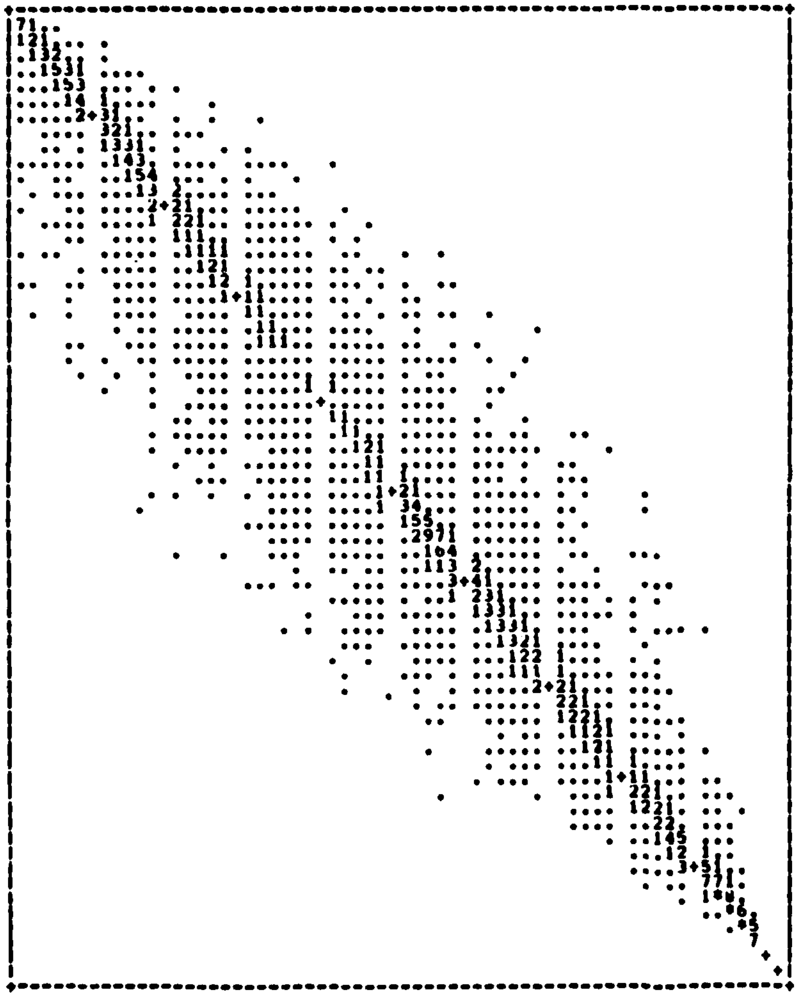
IMAGE1: GIRL 8x8, TYPE: IMAGE, BITS/PIXEL: 6.00000
VS
IMAGE2: GIRL 8x8, TYPE: DMAPS, BITS/PIXEL: 2.50000

AMPLIFIED DIFFERENCE IMAGE PRODUCED: 240 SUBFRAMES, AMPLIFICATION 6.0

DIFFERENCES, IMAGE1-IMAGE2: MEAN 0.000, RMS 12.317, MSE 0.66055%, RMSE 2.73859%

IMAGE 1 (UNIT = 24 PIXELS)

IMAGE 2



GRAY SCALE	IMAGE 1	IMAGE 2	DIFFERENCE	IMAGE1-IMAGE2
0 TO 3	243	216	-128 TO -125	0
4 TO 7	133	132	-124 TO -121	0
8 TO 11	192	212	-120 TO -117	0
12 TO 15	270	308	-116 TO -113	0
16 TO 19	306	292	-112 TO -109	0
20 TO 23	357	228	-108 TO -105	0
24 TO 27	0	236	-104 TO -101	0
28 TO 31	310	216	-100 TO -97	0
32 TO 35	265	240	-96 TO -93	0
36 TO 39	333	292	-92 TO -89	0
40 TO 43	339	328	-88 TO -85	3
44 TO 47	331	236	-84 TO -81	0
48 TO 51	0	216	-80 TO -77	1
52 TO 55	251	220	-76 TO -73	0
56 TO 59	235	184	-72 TO -69	8
60 TO 63	197	156	-68 TO -65	6
64 TO 67	219	216	-64 TO -61	11
68 TO 71	239	200	-60 TO -57	10
72 TO 75	0	184	-56 TO -53	15
76 TO 79	201	180	-52 TO -49	22
80 TO 83	187	148	-48 TO -45	35
84 TO 87	212	204	-44 TO -41	23
88 TO 91	176	184	-40 TO -37	45
92 TO 95	161	176	-36 TO -33	63
96 TO 99	183	192	-32 TO -29	78
100 TO 103	0	152	-28 TO -25	92
104 TO 107	194	168	-24 TO -21	110
108 TO 111	207	168	-20 TO -17	152
112 TO 115	196	228	-16 TO -13	226
116 TO 119	224	160	-12 TO -9	405
120 TO 123	235	208	-8 TO -5	799
124 TO 127	0	212	-4 TO -1	1609
128 TO 131	308	296	0 TO 3	7705
132 TO 135	406	404	4 TO 7	1786
136 TO 139	513	556	8 TO 11	749
140 TO 143	488	380	12 TO 15	417
144 TO 147	441	280	16 TO 19	244
148 TO 151	0	352	20 TO 23	155
152 TO 155	341	244	24 TO 27	118
156 TO 159	331	308	28 TO 31	92
160 TO 163	290	304	32 TO 35	85
164 TO 167	331	260	36 TO 39	43
168 TO 171	280	272	40 TO 43	44
172 TO 175	275	204	44 TO 47	31
176 TO 179	0	224	48 TO 51	20
180 TO 183	300	224	52 TO 55	20
184 TO 187	277	240	56 TO 59	19
188 TO 191	212	196	60 TO 63	9
192 TO 195	240	180	64 TO 67	9
196 TO 199	230	160	68 TO 71	10
200 TO 203	0	164	72 TO 75	3
204 TO 207	226	224	76 TO 79	4
208 TO 211	246	220	80 TO 83	3
212 TO 215	257	212	84 TO 87	0
216 TO 219	297	336	88 TO 91	1
220 TO 223	330	144	92 TO 95	0
224 TO 227	0	300	96 TO 99	0
228 TO 231	419	400	100 TO 103	0
232 TO 235	537	512	104 TO 107	0
236 TO 239	538	488	108 TO 111	0
240 TO 243	541	484	112 TO 115	0
244 TO 247	310	180	116 TO 119	0
248 TO 251	0	0	120 TO 123	0
252 TO 255	0	0	124 TO 127	0

Figure 9-4. Error Histogram For 2x2 MEAN Example

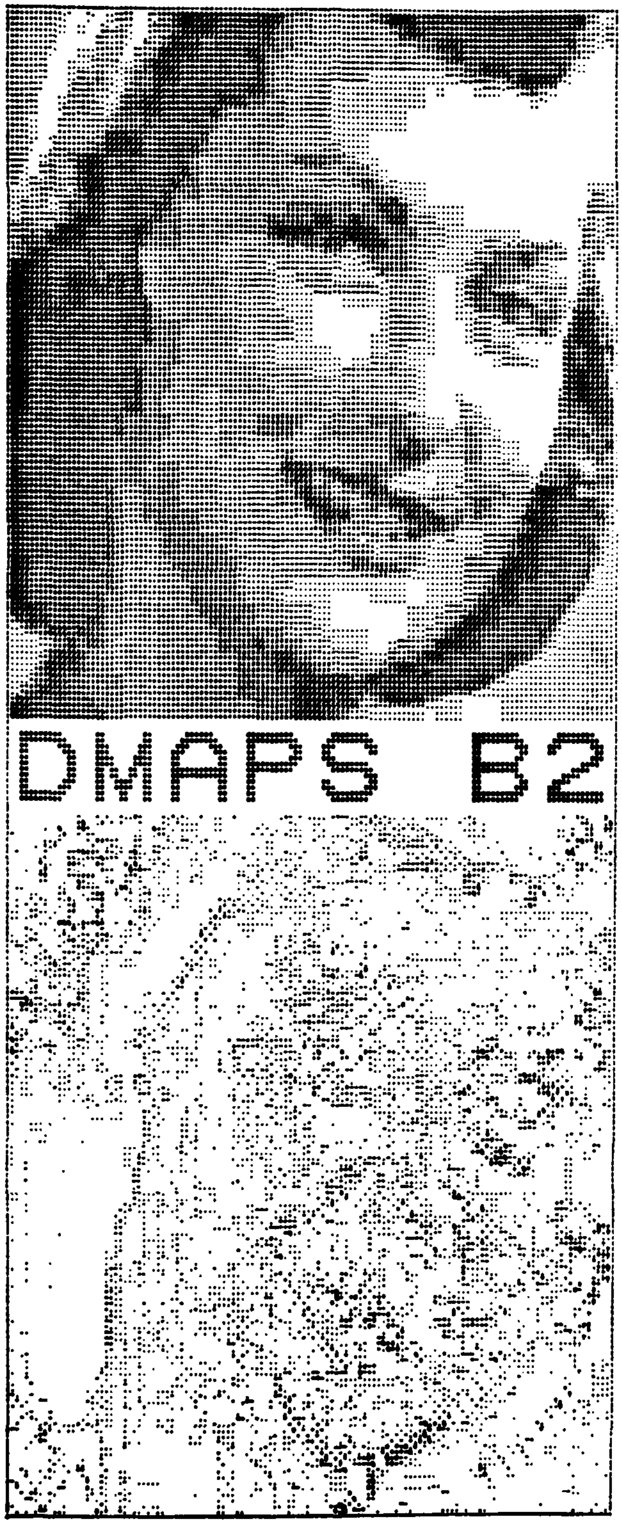


Figure 9-5. MAPS Block Mode at 2 bits/pixel

DIFFERENCE STATISTICS:

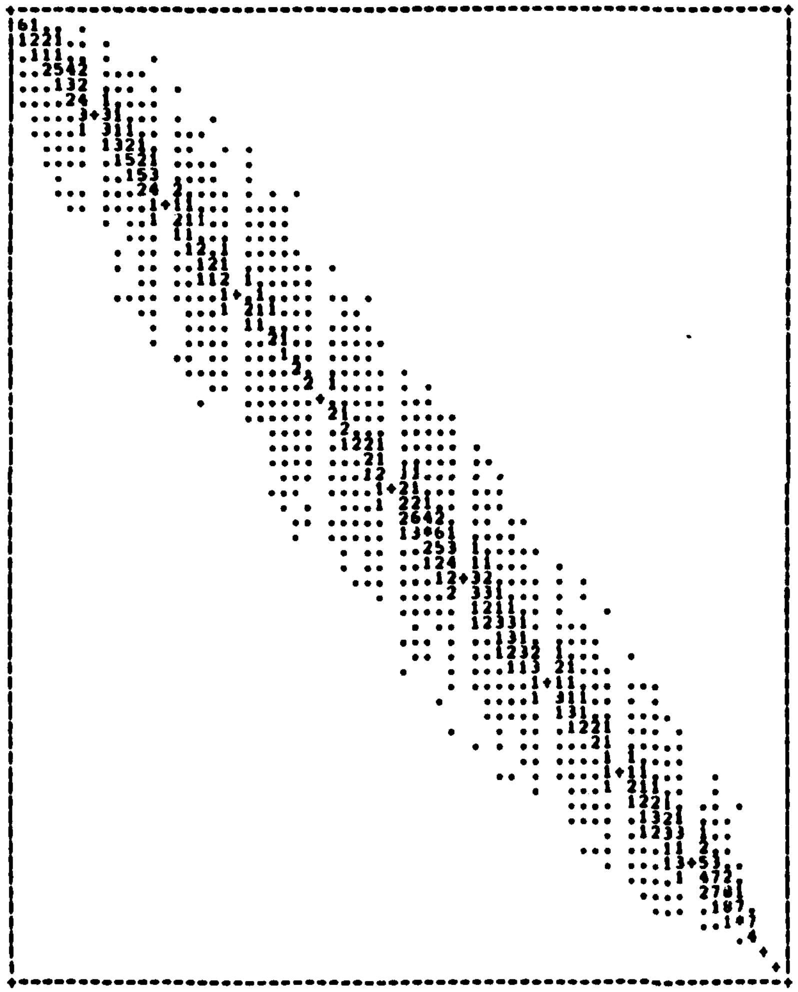
IMAGE1: GIRL 8X8, TYPE: IMAGE, BITS/PIXEL: 6.00000
VS
IMAGE2: GIRL 8X8, TYPE: DMAPS, BITS/PIXEL: 1.98854

AMPLIFIED DIFFERENCE IMAGE PRODUCED: 240 SUBFRAMES, AMPLIFICATION 6.0

DIFFERENCES, IMAGE1-IMAGE2: MEAN -0.081, RMS 8.374, MSE 0.30531%, RMSE 1.26501%

IMAGE 1 (UNIT = 24 PIXELS)

IMAGE 2



DISTRIBUTIONS

GRAY SCALE	IMAGE 1	IMAGE 2	DIFFERENCE	IMAGE1-IMAGE2
0 TO 3	243	204	-128 TO -125	0
4 TO 7	133	192	-124 TO -121	0
8 TO 11	192	153	-120 TO -117	0
12 TO 15	270	374	-116 TO -113	0
16 TO 19	306	236	-112 TO -109	0
20 TO 23	357	269	-108 TO -105	0
24 TO 27	0	272	-104 TO -101	0
28 TO 31	310	215	-100 TO -97	0
32 TO 35	265	241	-96 TO -93	0
36 TO 39	333	312	-92 TO -89	0
40 TO 43	339	300	-88 TO -85	0
44 TO 47	331	291	-84 TO -81	0
48 TO 51	0	188	-80 TO -77	0
52 TO 55	251	204	-76 TO -73	0
56 TO 59	235	130	-72 TO -69	0
60 TO 63	197	174	-68 TO -65	0
64 TO 67	219	217	-64 TO -61	0
68 TO 71	239	225	-60 TO -57	0
72 TO 75	0	164	-56 TO -53	0
76 TO 79	201	193	-52 TO -49	0
80 TO 83	187	155	-48 TO -45	0
84 TO 87	212	174	-44 TO -41	3
88 TO 91	176	98	-40 TO -37	9
92 TO 95	161	172	-36 TO -33	13
96 TO 99	183	200	-32 TO -29	36
100 TO 103	0	124	-28 TO -25	55
104 TO 107	194	158	-24 TO -21	72
108 TO 111	207	122	-20 TO -17	127
112 TO 115	196	244	-16 TO -13	256
116 TO 119	224	181	-12 TO -9	484
120 TO 123	235	208	-8 TO -5	1016
124 TO 127	0	176	-4 TO -1	1963
128 TO 131	308	216	0 TO 3	7354
132 TO 135	406	431	4 TO 7	1994
136 TO 139	513	610	8 TO 11	949
140 TO 143	488	338	12 TO 15	479
144 TO 147	441	317	16 TO 19	252
148 TO 151	0	292	20 TO 23	129
152 TO 155	341	311	24 TO 27	64
156 TO 159	331	227	28 TO 31	53
160 TO 163	290	323	32 TO 35	35
164 TO 167	331	227	36 TO 39	12
168 TO 171	280	314	40 TO 43	4
172 TO 175	275	257	44 TO 47	1
176 TO 179	0	160	48 TO 51	0
180 TO 183	300	238	52 TO 55	0
184 TO 187	277	233	56 TO 59	0
188 TO 191	212	249	60 TO 63	0
192 TO 195	240	186	64 TO 67	0
196 TO 199	230	139	68 TO 71	0
200 TO 203	0	152	72 TO 75	0
204 TO 207	226	199	76 TO 79	0
208 TO 211	246	242	80 TO 83	0
212 TO 215	257	247	84 TO 87	0
216 TO 219	297	319	88 TO 91	0
220 TO 223	330	188	92 TO 95	0
224 TO 227	0	360	96 TO 99	0
228 TO 231	419	397	100 TO 103	0
232 TO 235	537	516	104 TO 107	0
236 TO 239	538	478	108 TO 111	0
240 TO 243	541	512	112 TO 115	0
244 TO 247	310	116	116 TO 119	0
248 TO 251	0	0	120 TO 123	0
252 TO 255	0	0	124 TO 127	0

Figure 9-6. Error Histogram for DMAPS B2 Block Mode Example

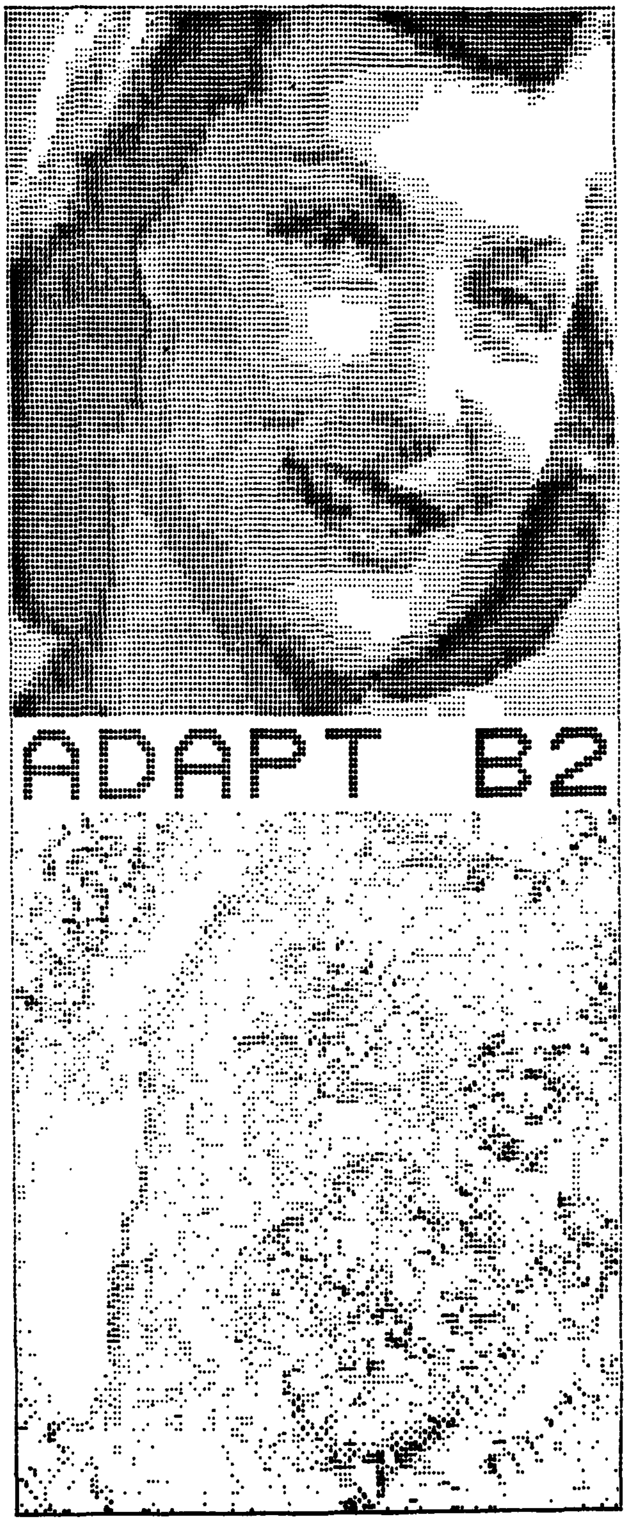


Figure 9-7. MAPS Block Mode With Adaptive Smoothing (Uniform Weight)

DIFFERENCE STATISTICS:

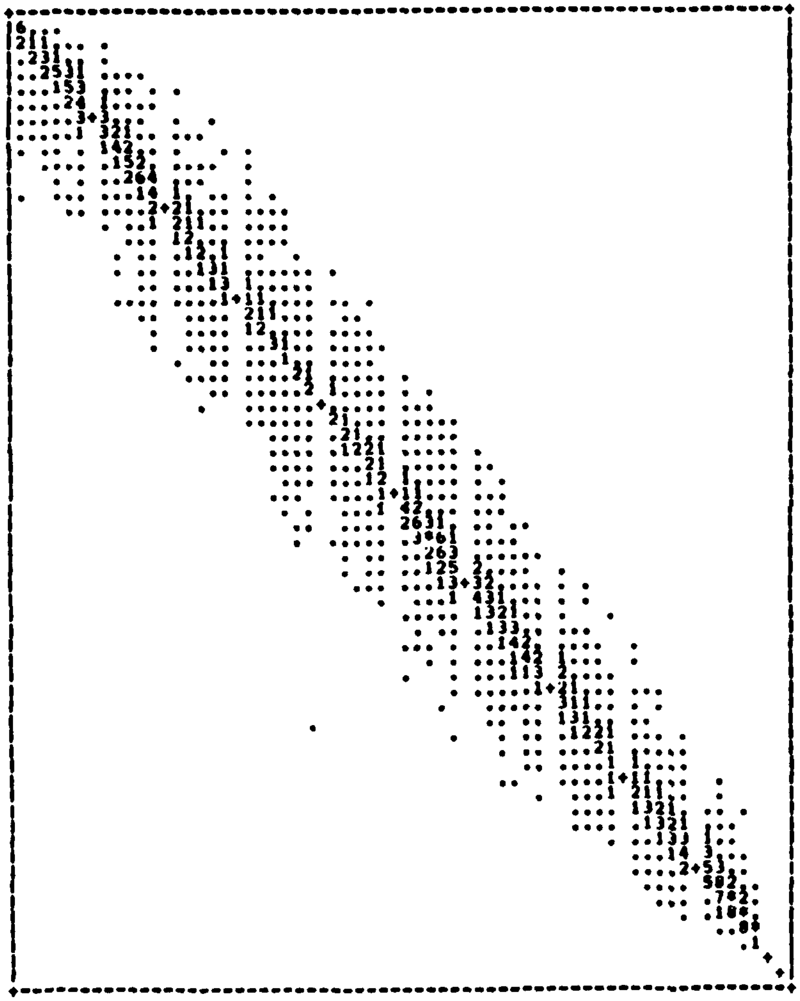
IMAGE1: GIRL 8X8, TYPE: IMAGE, BITS/PIXEL: 6.00000
VS
IMAGE2: GIRL 8X8, TYPE: ADAPT, BITS/PIXEL: 1.98854

AMPLIFIED DIFFERENCE IMAGE PRODUCED: 240 SUBFRAMES, AMPLIFICATION 6.0

DIFFERENCES, IMAGE1-IMAGE2: MEAN -0.016, RMS 7.998, MSE 0.278578, RMSE 1.154928

IMAGE 1 (UNIT = 24 PIXELS)

IMAGE 2



GRAY SCALE	DISTRIBUTIONS IMAGE 1	IMAGE 2	DIFFERENCE	IMAGE1-IMAGE2
0 TO 3	243	170	-128 TO -125	0
4 TO 7	133	136	-124 TO -121	0
8 TO 11	192	215	-120 TO -117	0
12 TO 15	270	351	-116 TO -113	0
16 TO 19	306	295	-112 TO -109	0
20 TO 23	357	246	-108 TO -105	0
24 TO 27	0	237	-104 TO -101	0
28 TO 31	310	220	-100 TO -97	0
32 TO 35	265	267	-96 TO -93	0
36 TO 39	333	282	-92 TO -89	0
40 TO 43	339	348	-88 TO -85	0
44 TO 47	331	252	-84 TO -81	0
48 TO 51	0	210	-80 TO -77	0
52 TO 55	251	205	-76 TO -73	0
56 TO 59	235	139	-72 TO -69	0
60 TO 63	197	170	-68 TO -65	0
64 TO 67	219	215	-64 TO -61	0
68 TO 71	239	216	-60 TO -57	0
72 TO 75	0	166	-56 TO -53	0
76 TO 79	201	182	-52 TO -49	0
80 TO 83	187	159	-48 TO -45	0
84 TO 87	212	203	-44 TO -41	4
88 TO 91	176	107	-40 TO -37	8
92 TO 95	161	172	-36 TO -33	12
96 TO 99	183	195	-32 TO -29	35
100 TO 103	0	121	-28 TO -25	55
104 TO 107	194	156	-24 TO -21	65
108 TO 111	207	140	-20 TO -17	118
112 TO 115	196	245	-16 TO -13	223
116 TO 119	224	165	-12 TO -9	395
120 TO 123	235	182	-8 TO -5	883
124 TO 127	0	166	-4 TO -1	1900
128 TO 131	308	265	0 TO 3	7970
132 TO 135	406	364	4 TO 7	1959
136 TO 139	513	621	8 TO 11	807
140 TO 143	488	342	12 TO 15	412
144 TO 147	441	341	16 TO 19	230
148 TO 151	0	326	20 TO 23	121
152 TO 155	341	292	24 TO 27	62
156 TO 159	331	257	28 TO 31	50
160 TO 163	290	281	32 TO 35	34
164 TO 167	331	265	36 TO 39	13
168 TO 171	280	297	40 TO 43	4
172 TO 175	275	246	44 TO 47	0
176 TO 179	0	209	48 TO 51	0
180 TO 183	300	218	52 TO 55	0
184 TO 187	277	234	56 TO 59	0
188 TO 191	212	228	60 TO 63	0
192 TO 195	240	184	64 TO 67	0
196 TO 199	230	154	68 TO 71	0
200 TO 203	0	155	72 TO 75	0
204 TO 207	226	204	76 TO 79	0
208 TO 211	246	246	80 TO 83	0
212 TO 215	257	243	84 TO 87	0
216 TO 219	297	288	88 TO 91	0
220 TO 223	330	267	92 TO 95	0
224 TO 227	0	336	96 TO 99	0
228 TO 231	419	450	100 TO 103	0
232 TO 235	537	514	104 TO 107	0
236 TO 239	538	497	108 TO 111	0
240 TO 243	541	470	112 TO 115	0
244 TO 247	310	33	116 TO 119	0
248 TO 251	0	0	120 TO 123	0
252 TO 255	0	0	124 TO 127	0

Figure 9-8. Error Histogram for ADAPT B2 Block Mode Example

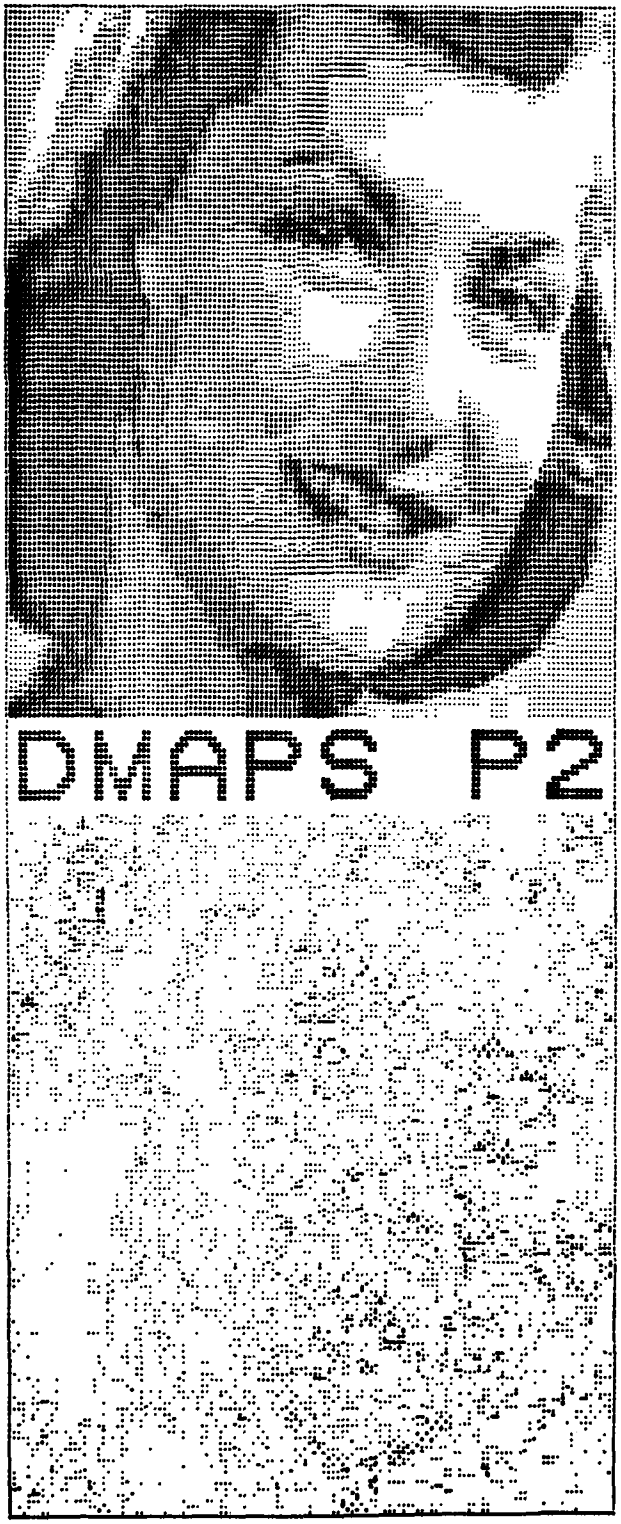


Figure 9-9. MAPS Pattern Mode at 2 bits/pixel

DIFFERENCE STATISTICS:

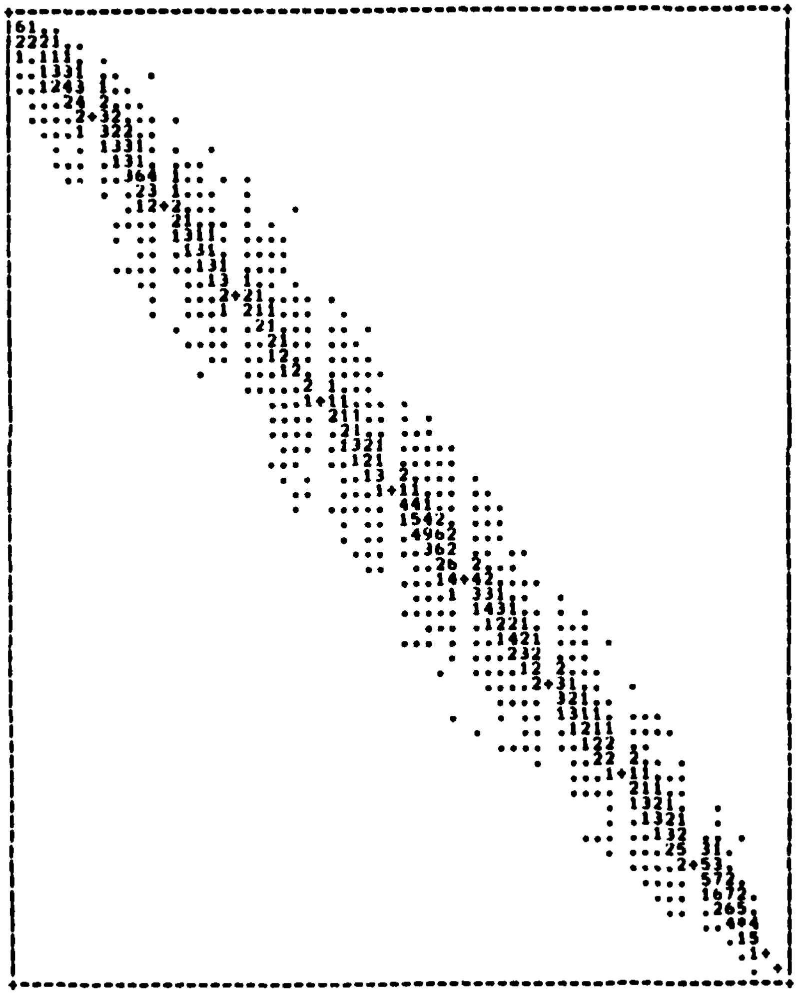
IMAGE1: GIRL 8x8, TYPE: IMAGE, BITS/PIXEL: 6.00000
VS
IMAGE2: GIRL 8x8, TYPE: DMAPS, BITS/PIXEL: 1.98854

AMPLIFIED DIFFERENCE IMAGE PRODUCED: 240 SUBFRAMES, AMPLIFICATION 6.0

DIFFERENCES, IMAGE1-IMAGE2: MEAN -0.107, RMS 6.433, MSE 0.180198, RMSE 0.747068

IMAGE 1 (UNIT = 24 PIXELS)

IMAGE 2



DISTRIBUTIONS

GRAY SCALE	IMAGE 1	IMAGE 2	DIFFERENCE	IMAGE1-IMAGE2
0 TO 3	243	202	-128 TO -125	0
4 TO 7	133	216	-124 TO -121	0
8 TO 11	192	179	-120 TO -117	0
12 TO 15	270	266	-116 TO -113	0
16 TO 19	306	338	-112 TO -109	0
20 TO 23	357	241	-108 TO -105	0
24 TO 27	0	254	-104 TO -101	0
28 TO 31	310	253	-100 TO -97	0
32 TO 35	265	255	-96 TO -93	0
36 TO 39	333	212	-92 TO -89	0
40 TO 43	339	406	-88 TO -85	0
44 TO 47	331	207	-84 TO -81	0
48 TO 51	0	190	-80 TO -77	0
52 TO 55	251	170	-76 TO -73	0
56 TO 59	235	208	-72 TO -69	0
60 TO 63	197	206	-68 TO -65	0
64 TO 67	219	187	-64 TO -61	0
68 TO 71	239	175	-60 TO -57	0
72 TO 75	0	192	-56 TO -53	0
76 TO 79	201	193	-52 TO -49	0
80 TO 83	187	137	-48 TO -45	0
84 TO 87	212	174	-44 TO -41	0
88 TO 91	176	160	-40 TO -37	1
92 TO 95	161	164	-36 TO -33	3
96 TO 99	183	136	-32 TO -29	9
100 TO 103	0	130	-28 TO -25	17
104 TO 107	194	194	-24 TO -21	38
108 TO 111	207	138	-20 TO -17	68
112 TO 115	196	232	-16 TO -13	108
116 TO 119	224	155	-12 TO -9	248
120 TO 123	235	228	-8 TO -5	903
124 TO 127	0	166	-4 TO -1	2475
128 TO 131	308	308	0 TO 3	8394
132 TO 135	406	347	4 TO 7	2066
136 TO 139	513	586	8 TO 11	647
140 TO 143	488	340	12 TO 15	197
144 TO 147	441	315	16 TO 19	87
148 TO 151	0	330	20 TO 23	52
152 TO 155	341	255	24 TO 27	28
156 TO 159	331	333	28 TO 31	16
160 TO 163	290	223	32 TO 35	2
164 TO 167	331	287	36 TO 39	0
168 TO 171	280	264	40 TO 43	1
172 TO 175	275	227	44 TO 47	0
176 TO 179	0	250	48 TO 51	0
180 TO 183	300	204	52 TO 55	0
184 TO 187	277	231	56 TO 59	0
188 TO 191	212	189	60 TO 63	0
192 TO 195	240	206	64 TO 67	0
196 TO 199	230	199	68 TO 71	0
200 TO 203	0	168	72 TO 75	0
204 TO 207	226	177	76 TO 79	0
208 TO 211	246	214	80 TO 83	0
212 TO 215	257	255	84 TO 87	0
216 TO 219	297	229	88 TO 91	0
220 TO 223	330	342	92 TO 95	0
224 TO 227	0	318	96 TO 99	0
228 TO 231	419	411	100 TO 103	0
232 TO 235	537	444	104 TO 107	0
236 TO 239	538	374	108 TO 111	0
240 TO 243	541	554	112 TO 115	0
244 TO 247	310	174	116 TO 119	0
248 TO 251	0	40	120 TO 123	0
252 TO 255	0	2	124 TO 127	0

Figure 9-10. Error Histogram For DMAPS P2 Pattern Mode Example

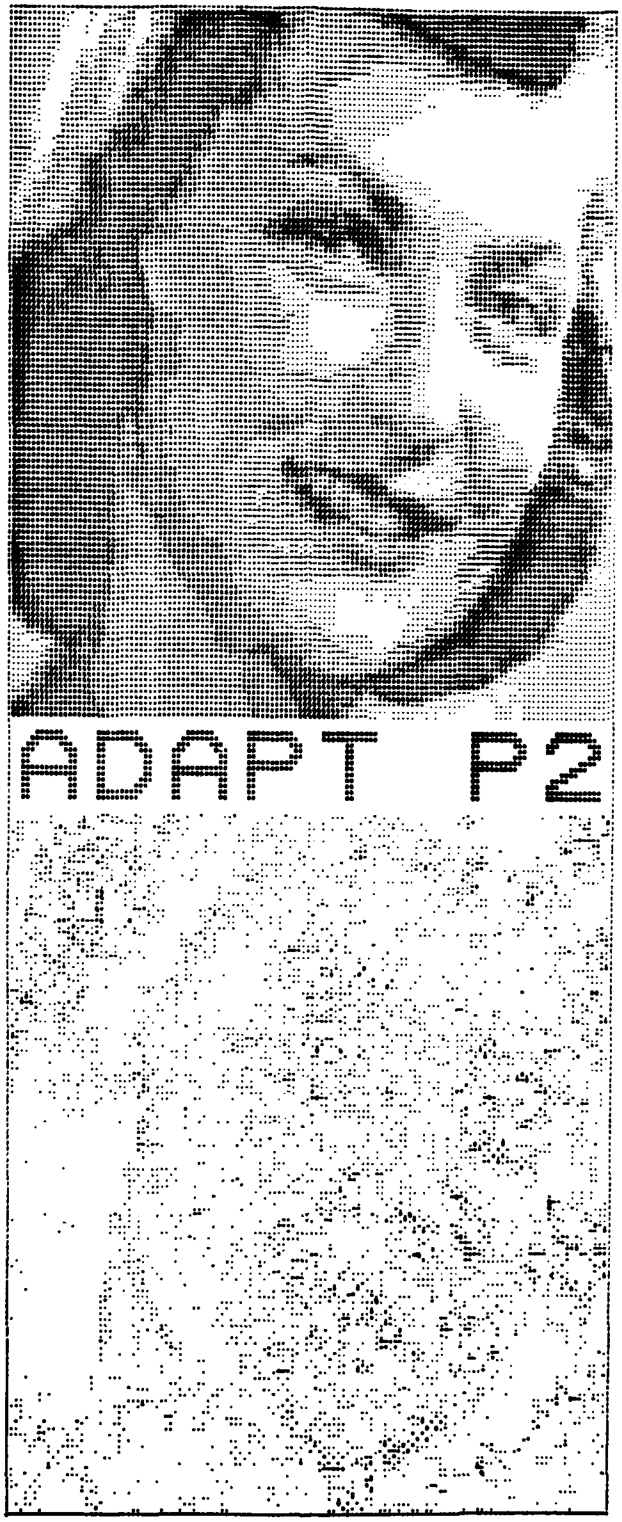


Figure 9-11. MAPS Pattern Mode With Adaptive Smoothing
(Uniform Weight)

DIFFERENCE STATISTICS:

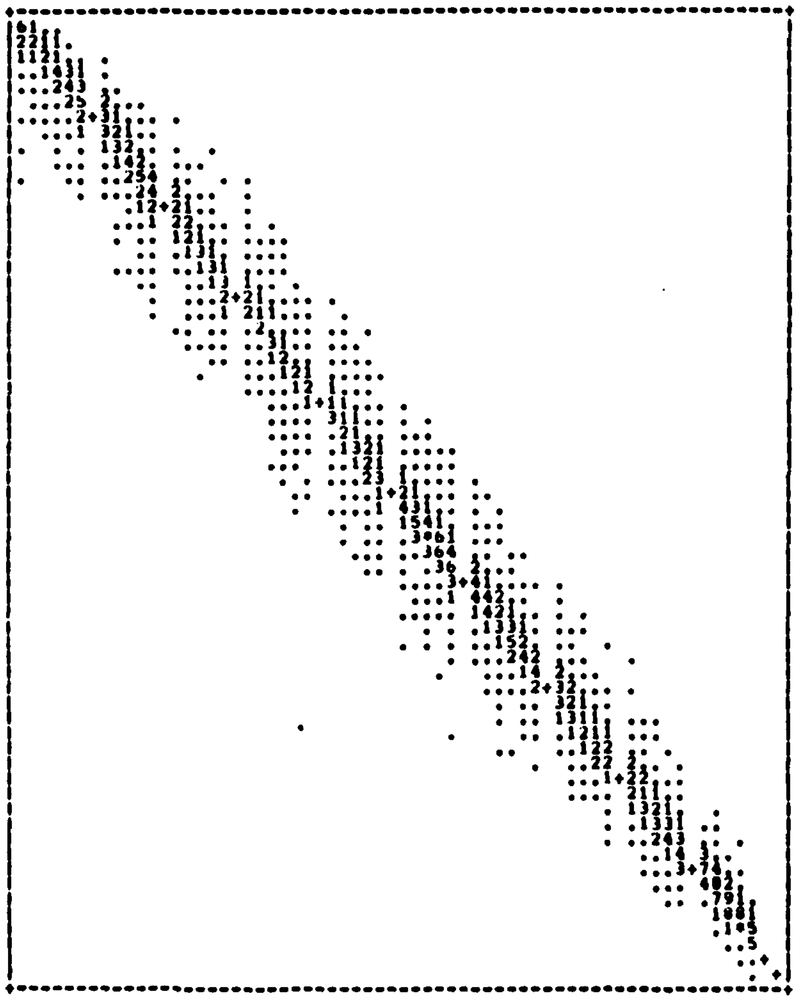
IMAGE1: GIRL 8x8, TYPE: IMAGE, BITS/PIXEL: 6.00000
VS
IMAGE2: GIRL 8x8, TYPE: ADAPT, BITS/PIXEL: 1.98854

AMPLIFIED DIFFERENCE IMAGE PRODUCED: 240 SUBFRAMES, AMPLIFICATION 6.0

DIFFERENCES, IMAGE1-IMAGE2: MEAN -0.021, RMS 6.173, MSE 0.16595%, RMSE 0.68802%

IMAGE 1 (UNIT = 24 PIXELS)

IMAGE 2



		DISTRIBUTIONS		
GRAY SCALE	IMAGE 1	IMAGE 2	DIFFERENCE	IMAGE1-IMAGE2
0 TO 3	243	193	-128 TO -125	0
4 TO 7	133	177	-124 TO -121	0
8 TO 11	192	201	-120 TO -117	0
12 TO 15	270	290	-116 TO -113	0
16 TO 19	306	296	-112 TO -109	0
20 TO 23	357	286	-108 TO -105	0
24 TO 27	0	245	-104 TO -101	0
28 TO 31	310	243	-100 TO -97	0
32 TO 35	265	236	-96 TO -93	0
36 TO 39	333	262	-92 TO -89	0
40 TO 43	339	337	-88 TO -85	0
44 TO 47	331	266	-84 TO -81	0
48 TO 51	0	167	-80 TO -77	0
52 TO 55	251	196	-76 TO -73	0
56 TO 59	235	199	-72 TO -69	0
60 TO 63	197	207	-68 TO -65	0
64 TO 67	219	166	-64 TO -61	0
68 TO 71	239	189	-60 TO -57	0
72 TO 75	0	188	-56 TO -53	0
76 TO 79	201	194	-52 TO -49	0
80 TO 83	187	140	-48 TO -45	0
84 TO 87	212	178	-44 TO -41	1
88 TO 91	176	168	-40 TO -37	1
92 TO 95	161	168	-36 TO -33	2
96 TO 99	183	135	-32 TO -29	9
100 TO 103	0	130	-28 TO -25	15
104 TO 107	194	201	-24 TO -21	40
108 TO 111	207	143	-20 TO -17	56
112 TO 115	196	212	-16 TO -13	102
116 TO 119	224	154	-12 TO -9	229
120 TO 123	235	204	-8 TO -5	792
124 TO 127	0	178	-4 TO -1	2249
128 TO 131	308	289	0 TO 3	8539
132 TO 135	406	354	4 TO 7	2276
136 TO 139	513	560	8 TO 11	662
140 TO 143	488	398	12 TO 15	190
144 TO 147	441	344	16 TO 19	98
148 TO 151	0	266	20 TO 23	53
152 TO 155	341	339	24 TO 27	28
156 TO 159	331	273	28 TO 31	15
160 TO 163	290	261	32 TO 35	3
164 TO 167	331	251	36 TO 39	0
168 TO 171	280	274	40 TO 43	0
172 TO 175	275	261	44 TO 47	0
176 TO 179	0	228	48 TO 51	0
180 TO 183	300	235	52 TO 55	0
184 TO 187	277	201	56 TO 59	0
188 TO 191	212	186	60 TO 63	0
192 TO 195	240	209	64 TO 67	0
196 TO 199	230	197	68 TO 71	0
200 TO 203	0	172	72 TO 75	0
204 TO 207	226	180	76 TO 79	0
208 TO 211	246	222	80 TO 83	0
212 TO 215	257	238	84 TO 87	0
216 TO 219	297	265	88 TO 91	0
220 TO 223	330	281	92 TO 95	0
224 TO 227	0	398	96 TO 99	0
228 TO 231	419	398	100 TO 103	0
232 TO 235	537	461	104 TO 107	0
236 TO 239	538	445	108 TO 111	0
240 TO 243	541	445	112 TO 115	0
244 TO 247	310	162	116 TO 119	0
248 TO 251	0	16	120 TO 123	0
252 TO 255	0	2	124 TO 127	0

Figure 9-12. Error Histogram for ADAPT P2 Pattern Mode Example

Figure 9-13. MAPS Pattern Mode With Adaptive Smoothing (Gaussian Weight)

DIFFERENCE STATISTICS:

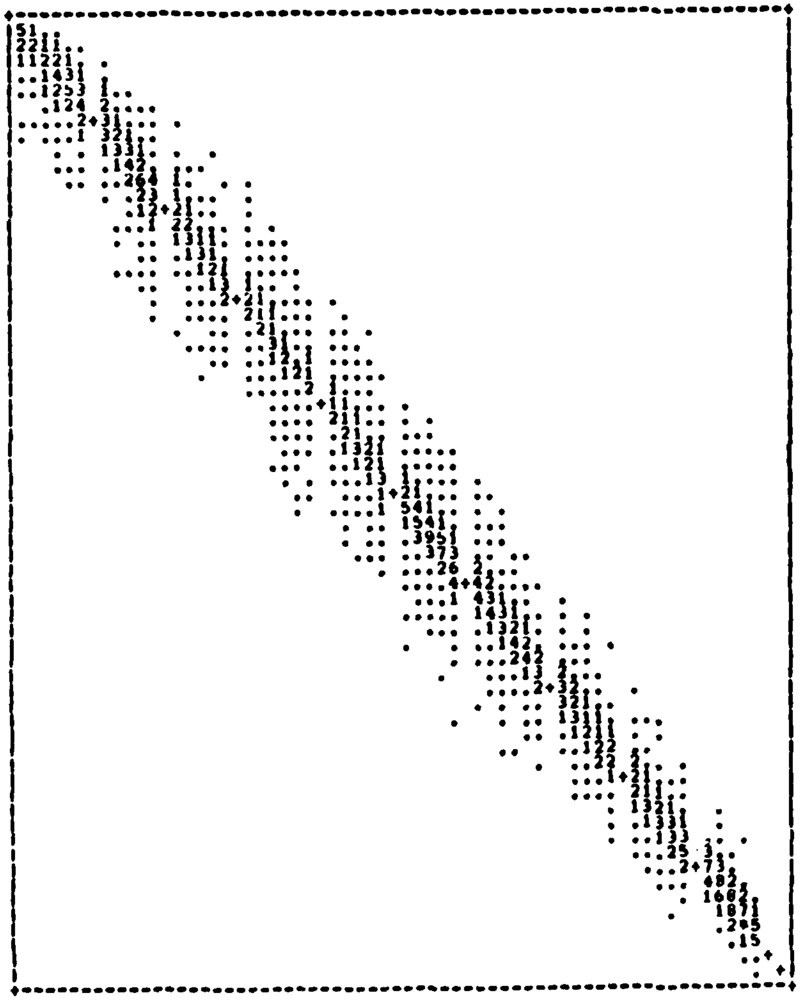
IMAGE1: GIRL 8x8, TYPE: IMAGE, BITS/PIXEL: 6.00000
VS
IMAGE2: GIRL 8x8, TYPE: ADAPT, BITS/PIXEL: 1.98854

AMPLIFIED DIFFERENCE IMAGE PRODUCED: 240 SUBFRAMES, AMPLIFICATION 6.0

DIFFERENCES, IMAGE1-IMAGE2: MEAN -0.051, RMS 6.093, MSE 0.161678, RMSE 0.670278

IMAGE 1 (UNIT = 24 PIXELS)

IMAGE 2



DISTRIBUTIONS

GRAY SCALE	IMAGE 1	IMAGE 2	DIFFERENCE	IMAGE1-IMAGE2
0 TO 3	243	190	-128 TO -125	0
4 TO 7	133	192	-124 TO -121	0
8 TO 11	192	218	-120 TO -117	0
12 TO 15	270	251	-116 TO -113	0
16 TO 19	306	329	-112 TO -109	0
20 TO 23	387	265	-108 TO -105	0
24 TO 27	0	242	-104 TO -101	0
28 TO 31	310	248	-100 TO -97	0
32 TO 35	265	247	-96 TO -93	0
36 TO 39	333	238	-92 TO -89	0
40 TO 43	339	374	-88 TO -85	0
44 TO 47	331	233	-84 TO -81	0
48 TO 51	0	178	-80 TO -77	0
52 TO 55	251	185	-76 TO -73	0
56 TO 59	235	206	-72 TO -69	0
60 TO 63	197	206	-68 TO -65	0
64 TO 67	219	179	-64 TO -61	0
68 TO 71	239	178	-60 TO -57	0
72 TO 75	0	191	-56 TO -53	0
76 TO 79	201	192	-52 TO -49	0
80 TO 83	187	137	-48 TO -45	0
84 TO 87	212	180	-44 TO -41	0
88 TO 91	176	169	-40 TO -37	1
92 TO 95	161	163	-36 TO -33	2
96 TO 99	183	132	-32 TO -29	8
100 TO 103	0	128	-28 TO -25	16
104 TO 107	194	195	-24 TO -21	35
108 TO 111	207	152	-20 TO -17	57
112 TO 115	196	217	-16 TO -13	98
116 TO 119	224	154	-12 TO -9	217
120 TO 123	235	212	-8 TO -5	818
124 TO 127	0	163	-4 TO -1	2326
128 TO 131	308	317	0 TO 3	8499
132 TO 135	406	344	4 TO 7	2266
136 TO 139	513	538	8 TO 11	650
140 TO 143	488	402	12 TO 15	190
144 TO 147	441	319	16 TO 19	83
148 TO 151	0	303	20 TO 23	61
152 TO 155	341	295	24 TO 27	26
156 TO 159	331	322	28 TO 31	15
160 TO 163	290	234	32 TO 35	2
164 TO 167	331	271	36 TO 39	0
168 TO 171	280	267	40 TO 43	0
172 TO 175	275	241	44 TO 47	0
176 TO 179	0	242	48 TO 51	0
180 TO 183	300	224	52 TO 55	0
184 TO 187	277	215	56 TO 59	0
188 TO 191	212	180	60 TO 63	0
192 TO 195	240	214	64 TO 67	0
196 TO 199	230	198	68 TO 71	0
200 TO 203	0	167	72 TO 75	0
204 TO 207	226	182	76 TO 79	0
208 TO 211	246	225	80 TO 83	0
212 TO 215	257	238	84 TO 87	0
216 TO 219	297	441	88 TO 91	0
220 TO 223	330	326	92 TO 95	0
224 TO 227	0	354	96 TO 99	0
228 TO 231	419	412	100 TO 103	0
232 TO 235	537	446	104 TO 107	0
236 TO 239	538	439	108 TO 111	0
240 TO 243	541	461	112 TO 115	0
244 TO 247	310	172	116 TO 119	0
248 TO 251	0	25	120 TO 123	0
252 TO 255	0	2	124 TO 127	0

Figure 9-14. Error Histogram for GAUSS P2 Pattern Mode Example

Figure 9-15. MAPS Pattern/Gaussian Mode With Dither (Amplitude = 8)

DIFFERENCE STATISTICS:

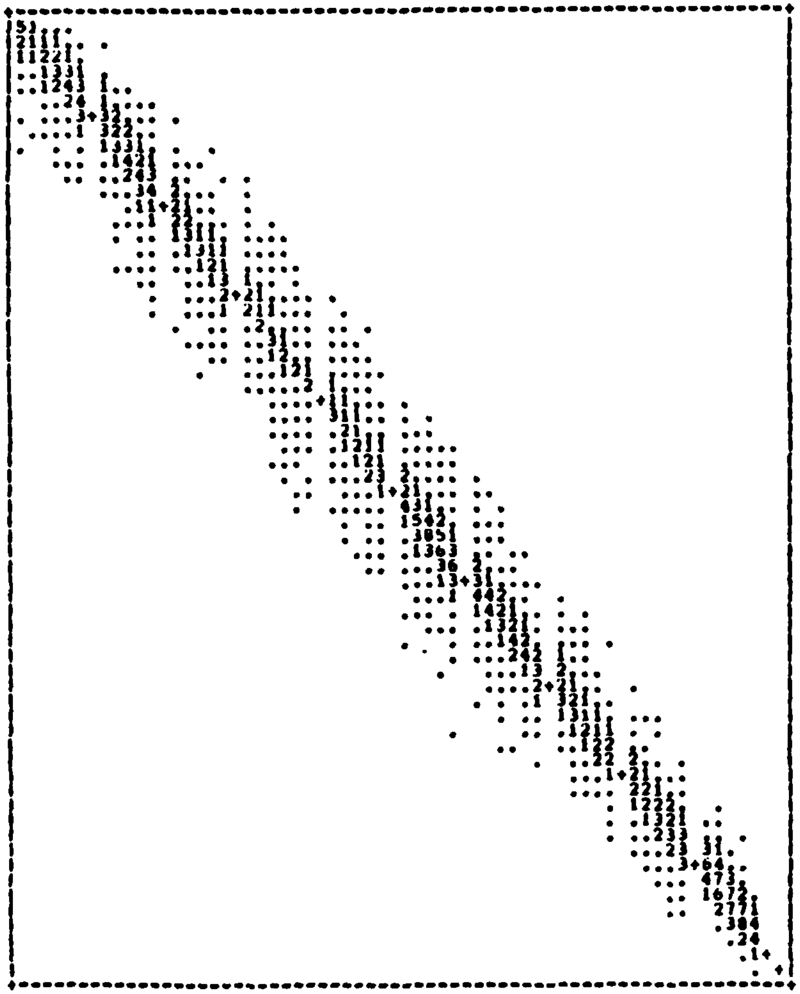
IMAGE1: GIRL 8x8, TYPE: IMAGE, BITS/PIXEL: 6.00000
VS
IMAGE2: GIRL 8x8, TYPE: ADAPT, BITS/PIXEL: 1.98854

AMPLIFIED DIFFERENCE IMAGE PRODUCED: 240 SUBFRAMES, AMPLIFICATION 6.0

DIFFERENCES, IMAGE1-IMAGE2: MEAN -0.050, RMS 6.349, MSE 0.175558, RMSE 0.727808

IMAGE 1 (UNIT = 24 PIXELS)

IMAGE 2



DISTRIBUTIONS

GRAY SCALE	IMAGE 1	IMAGE 2	DIFFERENCE	IMAGE1-IMAGE2
0 TO 3	243	200	-128 TO -125	0
4 TO 7	133	163	-124 TO -121	0
8 TO 11	192	247	-120 TO -117	0
12 TO 15	270	256	-116 TO -113	0
16 TO 19	306	299	-112 TO -109	0
20 TO 23	357	255	-108 TO -105	0
24 TO 27	0	275	-104 TO -101	0
28 TO 31	310	242	-100 TO -97	0
32 TO 35	265	246	-96 TO -93	0
36 TO 39	333	264	-92 TO -89	0
40 TO 43	339	298	-88 TO -85	0
44 TO 47	331	292	-84 TO -81	0
48 TO 51	0	162	-80 TO -77	0
52 TO 55	251	188	-76 TO -73	0
56 TO 59	235	210	-72 TO -69	0
60 TO 63	197	212	-68 TO -65	0
64 TO 67	219	162	-64 TO -61	0
68 TO 71	239	194	-60 TO -57	0
72 TO 75	0	179	-56 TO -53	0
76 TO 79	201	201	-52 TO -49	0
80 TO 83	167	130	-48 TO -45	0
84 TO 87	212	181	-44 TO -41	1
88 TO 91	176	164	-40 TO -37	0
92 TO 95	161	168	-36 TO -33	2
96 TO 99	183	137	-32 TO -29	9
100 TO 103	0	124	-28 TO -25	16
104 TO 107	194	200	-24 TO -21	35
108 TO 111	207	166	-20 TO -17	55
112 TO 115	196	182	-16 TO -13	109
116 TO 119	224	174	-12 TO -9	239
120 TO 123	235	210	-8 TO -5	893
124 TO 127	0	193	-4 TO -1	2511
128 TO 131	308	275	0 TO 3	7945
132 TO 135	406	384	4 TO 7	2374
136 TO 139	513	511	8 TO 11	780
140 TO 143	488	397	12 TO 15	203
144 TO 147	441	335	16 TO 19	94
148 TO 151	0	268	20 TO 23	49
152 TO 155	341	347	24 TO 27	29
156 TO 159	331	275	28 TO 31	14
160 TO 163	290	264	32 TO 35	2
164 TO 167	331	246	36 TO 39	0
168 TO 171	286	292	40 TO 43	0
172 TO 175	275	244	44 TO 47	0
176 TO 179	0	224	48 TO 51	0
180 TO 183	300	227	52 TO 55	0
184 TO 187	277	215	56 TO 59	0
188 TO 191	212	187	60 TO 63	0
192 TO 195	240	203	64 TO 67	0
196 TO 199	230	204	68 TO 71	0
200 TO 203	0	172	72 TO 75	0
204 TO 207	226	172	76 TO 79	0
208 TO 211	246	224	80 TO 83	0
212 TO 215	257	246	84 TO 87	0
216 TO 219	297	268	88 TO 91	0
220 TO 223	330	278	92 TO 95	0
224 TO 227	0	377	96 TO 99	0
228 TO 231	419	410	100 TO 103	0
232 TO 235	637	445	104 TO 107	0
236 TO 239	538	464	108 TO 111	0
240 TO 243	541	401	112 TO 115	0
244 TO 247	310	184	116 TO 119	0
248 TO 251	0	45	120 TO 123	0
252 TO 255	0	2	124 TO 127	0

Figure 9-16. Error Histogram For Pattern/Gaussian DITHER 8 Example

9.10 Mean Square Error Performance Comparisons

The qualitative image improvement described for the above sequence is also borne out by the quantitative fidelity measures. The histograms in Figures 9-4, 9-6, 9-8, 9-10, 9-12, and 9-14 exhibit progressive narrowing with a slight relaxation when random dither is added (Figure 9-16). The following tabulation of summary error statistics reflects this improvement as well:

Mean Square Error Performance:

Example	Mean Square Error	Relative MSE	RMS Error
2X2 MEAN	0.66055 %	2.73859 %	12.317
DMAPS B2	0.30531 %	1.26581 %	8.374
ADAPT B2	0.27857 %	1.15492 %	7.998
DMAPS P2	0.18019 %	0.74706 %	6.433
ADAPT P2	0.16595 %	0.68802 %	6.173
GAUSS P2	0.16167 %	0.67027 %	6.093
DITHER 8	0.17555 %	0.72780 %	6.349

From these summary statistics, it is seen that the 'block' mode is significantly better than simple averaging and the 'pattern' mode gives a further dramatic improvement. Although it is a heuristic technique, adaptive smoothing is also seen to yield a small but consistent enhancement in fidelity. Finally, the slight loss from the addition of dither seems more than compensated by its improvement of visual quality.

9.11 A 'Real'-Image Example

This closing subsection presents an example of the application of TransMAPS to a 'real' (video-sized) image - the BLDGIMG.DAT file which is supplied with the package. The interaction is summarized on the three planning forms displayed as Figures 9-17, 9-18, and 9-19. Not shown are two ANNOTE protocols, each of which assembles two image frames prior to the final four-frame composite. This is, however, an example of the recursive use of ANNOTE.

MAPS PLANNING FORM (COMPRESSION) (ALLOWED RANGE) [DEFAULT] DATE *March 1982*

MODULE | USER INPUTS

RUN SUBFRM ✓

FILES:
- IN - USER RASTER
- OUT - IMAGE.DAT

SOURCE IMAGE FILE NAME (*BLDGIMG*__)
USER IMAGE NAME (*BUILDING*)
SKIP LINES [0] *0* SKIP PIXELS [0] *0*
PROCESS LINES [480] *480* PROCESS PIXELS [624] *624*
BITS/PIXEL (6 8) [8]
SUBFRAME SIZE (8 16 32) [8] SUBFRAME GRID (SQUARE STAGGER) [SQUARE]
8X8 ONLY

RUN MAPS ✓

MODE (QUICK USER FULL) [Q]

FILES:
- IN - IMAGE.DAT
- OUT - MAPS.DAT
- IN/OUT - MSET.DAT USER PARAMETERS

MACRO FIDELITY CONTROL:
IMAGE PARTITION:
PIXEL INDEX DIRECTION →
LINE INDEX DIRECTION ↓
EACH CELL (1 2 3 4) [ALL 1]

MICRO FIDELITY CONTROL:

PARAMETERS	GROUP 1	GROUP 2	GROUP 3	GROUP 4
CONTRAST SCALE	30			
TAPER	[3.0]	[3.0]	[3.0]	[3.0]
STEP FRACTION	[0.5]	[0.5]	[0.5]	[0.5]
STEP BIAS	[0.1]	[0.1]	[0.1]	[0.1]

OR:

MATRIX	E	M	L	U	E	M	L	U	E	M	L	U	E	M	L	U
0-1																
1-2																
2-3																
3-4																
4-5																

BLOCK/PATTERN ASSIGNMENT, EACH LEVEL (B P) [BPPBBB]
012345 LEVEL

GRAY-SCALE MANIPULATIONS:

CONTRAST REMAP: CODE CONTRAST
PIECEWISE LINEAR BREAKPOINT PAIRS (NON-DECREASING)
CODE (0-255)
CONTRAST (0-4095)

CODE	CONTRAST
0	0
[255]	[255]
255	

INTENSITY REMAP: CODE INTENSITY
CODE (0-255)
INTENSITY (0-4095)

CODE	INTENSITY
0	0
[255]	[255]
255	

INTENSITY RESET:
(MEAN)
(PSEUDO-MEDIAN)
(LOWEST)
(SECOND)
(THIRD)
(HIGHEST)
[MEAN]

RESULTS: COMPRESSION: *1.99637 b/p*
MEAN SQUARE ERROR: *0.11911 %*

LEVEL:	0	1	2	3	4	5
MAPSELS:	*29720*	*20594*	*8646*	*835*		
OPTIMAL BIASES: -		*-2*	*0*			
+		*5*	*3*			

Figure 9-17. Compression Planning Form - Building Scene

MAPS PLANNING FORM (PRODUCT GENERATION) (ALLOWED RANGE) [DEFAULT]

DATE *March 1982*
IMAGE NAME *BUILDING*
COMPRESSION *1.39637*

MODULE — USER INPUTS

RUN DMAPS ✓

FILES:
- IN - MAPS.DAT
- OUT - DMAPS.DAT
- OUT - LEVEL.DAT

RUN ADAPT /NOT WITH STAGGER GRID/

FILES:
- IN - DMAPS.DAT
- IN - LEVEL.DAT
- OUT - ADAPT.DAT

CONVOLUTION WEIGHTING (GAUSSIAN UNIFORM) [G]
SIGMA MULTIPLE AT WINDOW CORNER [2.0] ____ /GAUSSIAN ONLY/
RANDOM DITHER AMPLITUDE [4.0] ____

RUN DIFFER ✓

FILES:
- IN - IMAGE.DAT, IN - DMAPS.DAT, IN - ADAPT.DAT } ANY TWO
- OUT - ERROR.DAT
- OUT - EPRINT.DAT (LISTING)

FILE PAIRINGS (IMAGE vs DMAPS IMAGE vs ADAPT DMAPS vs ADAPT) [I D]
DIFFERENCE PARAMETER:
(<0 SIGNED =0 STATISTICS ONLY >0 AMPLIFICATION FACTOR) [10] *10*

RUN RASTER ✓ *Three times*

FILES:
- IN - IMAGE.DAT, IN - DMAPS.DAT, IN - LEVEL.DAT, IN - ADAPT.DAT, IN - ERROR.DAT } ANY ONE
- OUT - IRAST.DAT, OUT - DRAST.DAT, OUT - LRAST.DAT, OUT - ARAST.DAT, OUT - ERAST.DAT } TYPE MATCHES INPUT

MAPS RASTER PRODUCT (IMAGE DMAPS *1st* LEVEL *2nd* ADAPT ERROR *3rd*) [DMAPS]

Figure 9-18. Product Generation Planning Form - Building Scene

ANNOTATION AND IMAGE ASSEMBLY PLANNING FORM

RUN ANNOTE

DATE March 1982

OUTPUT IMAGE:

TYPE (RASTER) FILE ANIMG.DAT   PRINTER LISTING) [R] APRINT.DAT   LINES 1080   PIXELS 1320   COMPLEMENT (Y (N)) [N]

EMBEDDED IMAGES: NUMBER (0 1 (2)) [0]

IMAGE	FILENAME	INPUT POSITION SKIP LINES	INPUT POSITION SKIP PIXELS	SIZE LINES	SIZE PIXELS	OUTPUT POSITION START LINE	OUTPUT POSITION START PIXEL	COMPLEMENT
1	TONAL	[0] 0	[0] 0	480	1280	21	21	(Y (N))
2	RESDIF	[0] 0	[0] 0	480	1280	541	21	(Y (N))

EMBEDDED MESSAGES: NUMBER (0 - 20) [0] 4

MSG	ORIENT	CHAR	SYMBOL SIZE	CENTER LINE	CENTER PIXEL	COMP	TEXT 1 5 10 15 20 25 30 35 40 45 50
1	((T)BLR)	11	(1(2)34)	518	332	(Y (N))	SOURCE 8B/P
2	((T)BLR)	9	(1(2)34)	518	988	(Y (N))	MAPS 2B/P
3	((T)BLR)	10	(1(2)34)	1038	332	(Y (N))	RESOLUTION
4	((T)BLR)	10	(1(2)34)	1038	988	(Y (N))	DIFFERENCE
5	(TBLR)		(1234)			(Y N)	
6	(TBLR)		(1234)			(Y N)	
7	(TBLR)		(1234)			(Y N)	
8	(TBLR)		(1234)			(Y N)	
9	(TBLR)		(1234)			(Y N)	
10	(TBLR)		(1234)			(Y N)	
11	(TBLR)		(1234)			(Y N)	
12	(TBLR)		(1234)			(Y N)	
13	(TBLR)		(1234)			(Y N)	
14	(TBLR)		(1234)			(Y N)	
15	(TBLR)		(1234)			(Y N)	
16	(TBLR)		(1234)			(Y N)	
17	(TBLR)		(1234)			(Y N)	
18	(TBLR)		(1234)			(Y N)	
19	(TBLR)		(1234)			(Y N)	
20	(TBLR)		(1234)			(Y N)	

Figure 9-19. Annotation Planning Form - Building Composite

Figure 9-20 presents the overall visual results. The frame contains the original source image, the MAPS decompressed image (at 2 bits/pixel), the resolution or level image, and an amplified difference image which has been complemented in ANNOTE so large differences are light in a dark frame. Finally, the difference statistics for this example are displayed as Figure 9-21.

Figure 9-20. 'Real'-Image MAPS Example-Building Scene

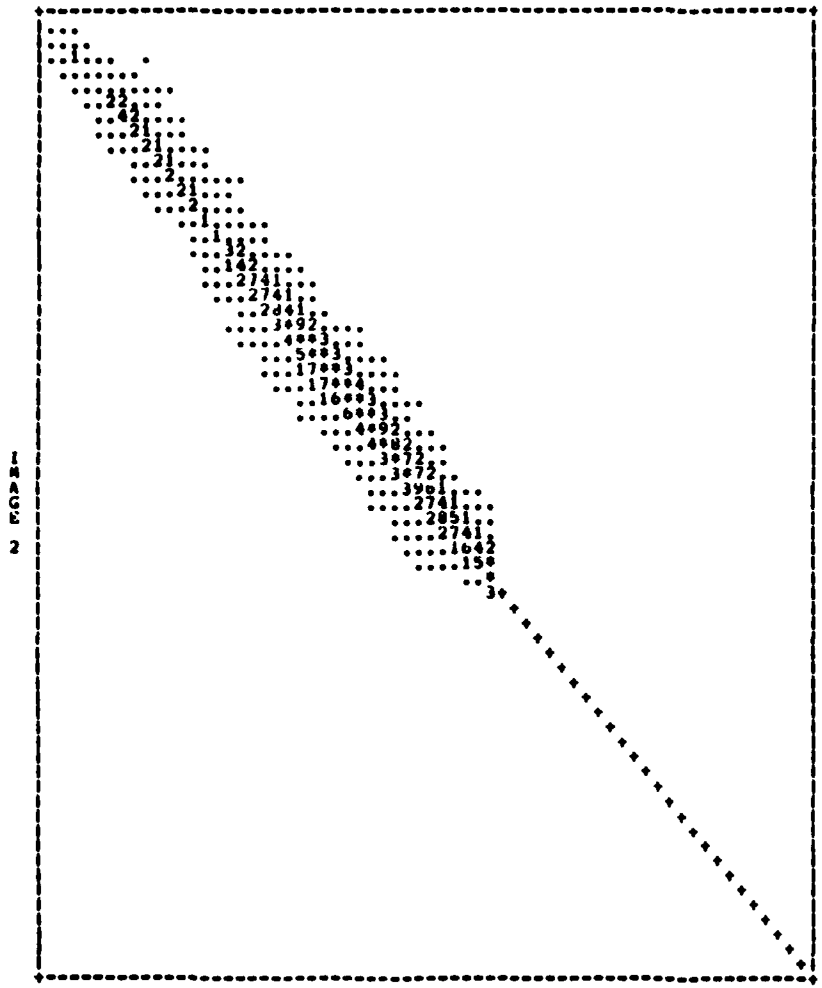
DIFFERENCE STATISTICS:

IMAGE1: BLDG HUM, TYPE: IMAGE, BITS/PIXEL: 8.00000
VS
IMAGE2: BLDG HOM, TYPE: DMAPS, BITS/PIXEL: 1.99637

AMPLIFIED DIFFERENCE IMAGE PRODUCED: 4748 SUBFRAMES, AMPLIFICATION 10.0

DIFFERENCES, IMAGE1-IMAGE2: MEAN -0.127, RMS 3.561, MSE 0.11910%, RMSE 1.48540%

IMAGE 1 (UNIT = 474 PIXELS)



DISTRIBUTIONS

GRAY SCALE	IMAGE 1	IMAGE 2	DIFFERENCE	IMAGE1-IMAGE2
0 TO 3	332	343	-128 TO -125	0
4 TO 7	351	326	-124 TO -121	0
8 TO 11	896	1304	-120 TO -117	0
12 TO 15	803	541	-116 TO -113	0
16 TO 19	783	723	-112 TO -109	0
20 TO 23	1824	2557	-108 TO -105	0
24 TO 27	3602	4163	-104 TO -101	0
28 TO 31	2938	2283	-100 TO -97	0
32 TO 35	1950	1766	-96 TO -93	0
36 TO 39	2021	2039	-92 TO -89	0
40 TO 43	2193	1996	-88 TO -85	0
44 TO 47	2108	2303	-84 TO -81	0
48 TO 51	1784	1510	-80 TO -77	0
52 TO 55	1358	1332	-76 TO -73	0
56 TO 59	1598	1561	-72 TO -69	0
60 TO 63	2649	3170	-68 TO -65	0
64 TO 67	4324	4467	-64 TO -61	0
68 TO 71	6338	7406	-60 TO -57	0
72 TO 75	7706	7956	-56 TO -53	0
76 TO 79	8827	8365	-52 TO -49	0
80 TO 83	11595	13765	-48 TO -45	0
84 TO 87	15909	16693	-44 TO -41	0
88 TO 91	19086	20190	-40 TO -37	0
92 TO 95	21302	22072	-36 TO -33	0
96 TO 99	20951	21286	-32 TO -29	0
100 TO 103	19877	19909	-28 TO -25	0
104 TO 107	18669	18726	-24 TO -21	0
108 TO 111	16755	15495	-20 TO -17	16
112 TO 115	14891	14160	-16 TO -13	70
116 TO 119	13313	12623	-12 TO -9	646
120 TO 123	12088	12111	-8 TO -5	6111
124 TO 127	10841	10306	-4 TO -1	42003
128 TO 131	9249	8238	0 TO 3	208462
132 TO 135	8510	8584	4 TO 7	39834
136 TO 139	7809	7886	8 TO 11	6087
140 TO 143	6794	7535	12 TO 15	574
144 TO 147	5841	8693	16 TO 19	61
148 TO 151	15807	7891	20 TO 23	8
152 TO 155	0	1598	24 TO 27	0
156 TO 159	0	0	28 TO 31	0
160 TO 163	0	0	32 TO 35	0
164 TO 167	0	0	36 TO 39	0
168 TO 171	0	0	40 TO 43	0
172 TO 175	0	0	44 TO 47	0
176 TO 179	0	0	48 TO 51	0
180 TO 183	0	0	52 TO 55	0
184 TO 187	0	0	56 TO 59	0
188 TO 191	0	0	60 TO 63	0
192 TO 195	0	0	64 TO 67	0
196 TO 199	0	0	68 TO 71	0
200 TO 203	0	0	72 TO 75	0
204 TO 207	0	0	76 TO 79	0
208 TO 211	0	0	80 TO 83	0
212 TO 215	0	0	84 TO 87	0
216 TO 219	0	0	88 TO 91	0
220 TO 223	0	0	92 TO 95	0
224 TO 227	0	0	96 TO 99	0
228 TO 231	0	0	100 TO 103	0
232 TO 235	0	0	104 TO 107	0
236 TO 239	0	0	108 TO 111	0
240 TO 243	0	0	112 TO 115	0
244 TO 247	0	0	116 TO 119	0
248 TO 251	0	0	120 TO 123	0
252 TO 255	0	0	124 TO 127	0

Figure 9-21. Error Histogram For Building Scene

APPENDIX

FUI'L SIZE TransMAPS PLANNING FORMS

(Suitable for Photocopying)

MAPS PLANNING FORM (COMPRESSION) (ALLOWED RANGE) [DEFAULT] DATE ________

MODULE — **USER INPUTS**

RUN SUBFRM

FILES:
- IN - USER RASTER
- OUT - IMAGE.DAT

SOURCE IMAGE FILE NAME (_ _ _ _ _ _ _ _ _ _ _ _ _)
USER IMAGE NAME (_ _ _ _ _ _ _ _ _ _)
SKIP LINES [0]________ SKIP PIXELS [0] ________
PROCESS LINES[ ] ________ PROCESS PIXELS[ ] ________
BITS/PIXEL (6 8) [8]
SUBFRAME SIZE (8 16 32) [8] SUBFRAME GRID (SQUARE STAGGER) [SQUARE]
8X8 ONLY

RUN MAPS

MODE (QUICK USER FULL) [Q]

FILES:
- IN - IMAGE.DAT
- OUT - MAPS.DAT
- IN/OUT - MSET.DAT USER PARAMETERS

MACRO FIDELITY CONTROL:

IMAGE PARTITION:

PIXEL INDEX DIRECTION →

LINE INDEX DIRECTION ↓

EACH CELL (1 2 3 4) [ALL 1]

MICRO FIDELITY CONTROL:

PARAMETERS	GROUP 1	GROUP 2	GROUP 3	GROUP 4
CONTRAST SCALE				
TAPER	[3.0]	[3.0]	[3.0]	[3.0]
STEP FRACTION	[0.5]	[0.5]	[0.5]	[0.5]
STEP BIAS	[0.1]	[0.1]	[0.1]	[0.1]

OR:

MATRIX	E	M	L	U	E	M	L	U	E	M	L	U	E	M	L	U
0-1																
1-2																
2-3																
3-4																
4-5																

BLOCK/PATTERN ASSIGNMENT, EACH LEVEL (B P) [BPPBBB]
012345 LEVEL

GRAY-SCALE MANIPULATIONS:

CONTRAST REMAP: CODE CONTRAST

	CODE	CONTRAST
	0	0
PIECEWISE LINEAR BREAKPOINT PAIRS	[255]	[255]
(NON-DECREASING)		
CODE (0-255)		
CONTRAST (0-4095)		
	255	

INTENSITY REMAP: CODE INTENSITY

	CODE	INTENSITY
	0	0
	[255]	[255]
CODE (0-255)		
INTENSITY (0-4095)		
	255	

INTENSITY RESET:
(MEAN)
(PSEUDO-MEDIAN)
(LOWEST)
(SECOND)
(THIRD)
(HIGHEST)
[MEAN]

RESULTS: COMPRESSION: ________
MEAN SQUARE ERROR: ________ %

LEVEL:	0	1	2	3	4	5
MAPSELS:						
OPTIMAL BIASES: -						
+						

MAPS PLANNING FORM (ALLOWED RANGE) [DEFAULT]

(PRODUCT GENERATION)

DATE________________

IMAGE NAME________________

COMPRESSION________________

MODULE | **USER INPUTS**

RUN DMAPS

FILES:

- IN - MAPS.DAT
- OUT - DMAPS.DAT
- OUT - LEVEL.DAT

RUN ADAPT /NOT WITH STAGGER GRID/

FILES:

- IN - DMAPS.DAT
- IN - LEVEL.DAT
- OUT - ADAPT.DAT

CONVOLUTION WEIGHTING (GAUSSIAN UNIFORM) [G]

SIGMA MULTIPLE AT WINDOW CORNER [2.0] ____ /GAUSSIAN ONLY/

RANDOM DITHER AMPLITUDE [4.0] ____

RUN DIFFER

FILES:

- IN - IMAGE.DAT
- IN - DMAPS.DAT
- IN - ADAPT.DAT

} ANY TWO

- OUT - ERROR.DAT
- OUT - EPRINT.DAT (LISTING)

FILE PAIRINGS (IMAGE vs DMAPS IMAGE vs ADAPT DMAPS vs ADAPT) [I D]

DIFFERENCE PARAMETER:

(<0 SIGNED =0 STATISTICS ONLY >0 AMPLIFICATION FACTOR) [10]____

RUN RASTER

FILES:

- IN - IMAGE.DAT
- IN - DMAPS.DAT
- IN - LEVEL.DAT
- IN - ADAPT.DAT
- IN - ERROR.DAT

} ANY ONE

- OUT - IRAST.DAT
- OUT - DRAST.DAT
- OUT - LRAST.DAT
- OUT - ARAST.DAT
- OUT - ERAST.DAT

} TYPE MATCHES INPUT

MAPS RASTER PRODUCT (IMAGE DMAPS LEVEL ADAPT ERROR) [DMAPS]

Annotation and Image Assembly Planning Form

RUN ANNOTE

DATE ____________

Output Image:

TYPE (RASTER FILE PRINTER LISTING) [R] LINES ________ PIXELS ________ COMPLEMENT (Y N) [N]
ANIMG.DAT APRINT.DAT

Embedded Images: NUMBER (0 1 2) [0]

IMAGE	FILENAME	INPUT POSITION SKIP LINES	INPUT POSITION SKIP PIXELS	SIZE LINES	SIZE PIXELS	OUTPUT POSITION START LINE	OUTPUT POSITION START PIXEL	COMPLEMENT
1	- - - - - - - - -	[0] ______	[0] ______	______	______	______	______	(Y N)
2	- - - - - - - - -	[0] ______	[0] ______	______	______	______	______	(Y N)

Embedded Messages: NUMBER (0 - 20) [0]

MSG	ORIENT	CHAR	SYMBOL SIZE	CENTER LINE	CENTER PIXEL	COMP	TEXT 1 5 10 15 20 25 30 35 40 45 50
1	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -
2	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -
3	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -
4	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -
5	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -
6	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -
7	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -
8	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -
9	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -
10	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -
11	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -
12	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -
13	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -
14	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -
15	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -
16	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -
17	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -
18	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -
19	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -
20	(TBLR)	___	(1234)	____	____	(Y N)	- - - - - - - - - -

PREFACE

This is Volume III of the Final Technical Report on Transportable MAPS Software. It constitutes the TransMAPS Program Maintenance Manual; its companion volumes contain a description of TransMAPS development (Volume I) and the TransMAPS User's Manual (Volume II). This volume is submitted in fulfillment of CDRL item A004 of Contract # F30602-80-C-0326.

TABLE OF CONTENTS

VOLUME III. TRANSMAPS MAINTENANCE MANUAL

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF TABLES

SECTION ONE

TRANSPORTABLE MAPS SOFTWARE: THE SYSTEMS VIEW

This document provides formal description of the Transportable MAPS Software Package or 'TransMAPS' from a systems viewpoint.

1.1 Purpose

Micro-Adaptive Picture Sequencing (MAPS) is a computationally-efficient, contrast-adaptive, variable-resolution spatial image coding technique. The TransMAPS Software Package implements the MAPS processes and related support functions in an integrated software system which is designed to be transportable among a variety of high-use mini-computers in the DEC computer family. The purpose of this implementation is to broaden access to MAPS. The ultimate intent is to establish a vehicle suitable for direct exploration of the MAPS technique and to provide a system capable of supporting functional application of MAPS to real image coding tasks.

The current document addresses those areas specifically concerned with the relation of TransMAPS to its host computer system. These include software installation, modification, and maintenance activities. The intended audience consists of systems personnel charged with support of TransMAPS at their site and others who wish to understand MAPS at a detailed level of 'how it's done'.

1.2 Maintenance Manual Organization

The information necessary for initial installation of TransMAPS at a new

site is given in Sections Two, Three, and Four. Section Two provides an overview of TransMAPS and the conditions assumed for the host computer environment. Section Three describes the extensive file system which supports TransMAPS and specifies the requirements imposed on communication of user image files. Section Four deals specifically with questions of TransMAPS installation from its source on computer-compatible magnetic tape.

Sections Five and Six discuss more details on the internal structure of the TransMAPS process modules. Section Five addresses questions of program modification which may arise in refining TransMAPS for a particular site. Section Six outlines general program structure conventions which should provide helpful guides in subsequent program maintenance activities.

Finally, Sections Seven through Thirteen contain complete COMMENT-annotated program listings of the TransMAPS source code. These listings have had running titles appended which give both main module and current-routine names. This should be an aid to rapid location of the relevant section for this deepest level of detailed documentation.

1.3 References

The material on initial installation in this document is intended as a self-contained path to achieving a functioning TransMAPS system. Beyond this however, at the level of program modification or maintenance, it is assumed that the reader is familiar with both the conceptual basis of MAPS and the structure of the user option space in the TransMAPS package. This material is described in the TransMAPS User's Manual to a level adequate for most purposes. For more detailed information on specific MAPS concepts and processes, the reader is referred to the following collection of documents:

References:

RADC MAPS-Related Reports:
---- ------------ --------

LaBonte, A. E. and C. J. McCallum (Control Data Corporation),
IMAGE COMPRESSION TECHNIQUES, RADC-TR-77-405, December 1977,
Final Technical Report, Contract No. F30602-76-C-0350.

LaBonte, A. E. and T. E. Rosenthal (Control Data Corporation),
MAPS IMAGE COMPRESSION, RADC-TR-80-173, May 1980,
Final Technical Report, Contract No. F30602-78-C-0253.

LaBonte, A. E. (Control Data Corporation),
INFRARED DATA COMPRESSION STUDY, RADC-TR-80-287, August 1980,
Final Technical Report, Contract No. F30602-79-C-0080.

SPIE Proceedings Articles:
---- ----------- --------

LaBonte, A. E., "Two-Dimensional Image Coding by
Micro-Adaptive Picture Sequencing (MAPS)",
Proceedings of the Society of Photo-Optical
Instrumentation Engineers, Volume 119,
APPLICATIONS OF DIGITAL IMAGE PROCESSING,
pp 99-106, 1977.

LaBonte, A. E., "Micro-Adaptive Picture Sequencing
(MAPS) in a Display Environment", Proceedings of
the Society of Photo-Optical Instrumentation Engineers,
Volume 249, ADVANCES IN IMAGE TRANSMISSION II,
pp 61-70, 1980.

SECTION TWO

TRANSMAPS OVERVIEW

This section outlines the contents of the TransMAPS package, its structure, and the assumed host computer environment.

2.1 The TransMAPS Package

The contents of the TransMAPS package are listed in Table 2-I.

The seven main program modules are provided on computer-compatible magnetic tape in the form of both source code and object code files. The first six modules relate directly to MAPS processes. Module #7 provides a stand-alone image assembly and annotation capability. This seventh module supports display of MAPS product images with assembly of comparative image frames and corresponding identifying annotation. It can, however, also be used with other user imagery to integrate and label related frames for various forms of presentation.

The six MAPS modules further subdivide into two functional classes. The first two relate to the MAPS compression phase. Module #1 converts the raster source imagery to the appropriate subframe organization for MAPS processing. Module #2 implements the actual MAPS micro-coding technique on a subframe by subframe basis. This second module contains the most complex portions of the interactive user option space.

The next four modules deal with MAPS product formation from the compressed image stream. Module #3 reconstructs the tonal image - MAPS decompression - and simultaneously generates a MAPS 'resolution image' which displays the micro-adaptive variable resolution created by MAPS.

TABLE 2-I. CONTENTS OF TransMAPS

```
TransMAPS Package:
--------- --------

        Seven Main Program Modules:
        ----- ---- ------- --------
                #1  SUBFRM  Raster to Subframe Conversion     -  SF.FSV, SF.OBJ
                #2  MAPS    MAPS Compression                  -  MP.FSV, MP.OBJ
                #3  DMAPS   MAPS Decompression & Level Image Formation
                                                              -  DM.FSV, DM.OBJ
                #4  ADAPT   MAPS Adaptive Image Smootning     -  AD.FSV, AD.OBJ
                #5  DIFFER  MAPS Difference Image Formation   -  DF.FSV, DF.OBJ
                #6  RASTER  Subframe to Raster Conversion     -  RS.FSV, RS.OBJ
                #7  ANNOTE  Image Assembly and Annotation     -  AI.FSV, AI.OBJ

                Provided on Computer-Compatible Tape (CCT):
                        FORTRAN IV-Plus Source Code  -  x.FSV
                        FORTRAN Object (F4P) Code  -  x.OBJ

        Six Data Sets:
        --- ---- -----
                Annotation Symbol-Map Tables                       -  SYMBOL.BIN
                MAPS Compression Test Image (160 x 128)            -  MTEST.BIN
                Sample MAPS User Parameter Set (Use with MTEST)    -  MSET.BIN
                MAPS Product Generation Test Image (120 x 128)     -  GIRL6.BIN
                Two 'Video Frame' Images (480 lines x 624 pixels):
                        Building Scene                             -  BLDGIMG.BIN
                        IEEE Girl                                  -  GIRLIMG.BIN

                Provided on Computer-Compatible Tape (CCT)

        MAPS File Structure with Standard Filenames and Headers:
        ---- ---- --------- ---- -------- --------- --- -------
                Source image (One Subframe/FIXED Record)      -  IMAGE.DAT
                User Parameter Set                            -  MSET.DAT
                MAPS Compression Stream (FIXED Records)       -  MAPS.DAT
                MAPS Block/Pattern Image (Subframes)          -  DMAPS.DAT
                MAPS Resolution Image (Subframes)             -  LEVEL.DAT
                MAPS Adaptively Smoothed Image (Subframes)    -  ADAPT.DAT
                MAPS Difference Image (Subframes)             -  ERROR.DAT
                Fidelity Performance Summary (Listing)        -  EPRINT.DAT
                MAPS Product Image (Raster)                   -  xRAST.DAT
                                                                 x = I,D,L,A,E
                Annotation Symbol-Map Tables                  -  SYMBOL.DAT
                Annotated Image (Raster)                      -  ANIMG.DAT
                Annotated Printer Pseudo-Image (Listing)      -  APRINT.DAT

        User Interaction Pre-Planning Aids:
        ---- ----------- ----------- -----
                MAPS Planning Form  -  Compression
                MAPS Planning Form  -  Product Generation
                Annotation and Image Assembly Planning Form

        Documentation:
        --------------
                TransMAPS User's Manual
                TransMAPS Maintenance Manual
```

Module #4 further refines the decompressed image through a MAPS-based adaptive smoothing process. Module #5 evaluates the fidelity of the MAPS coding through formation of a difference image between the source and product images. Finally, Module #6 converts any of the subframe organized MAPS images back to raster format for interface with the outside world.

The seven program modules are supported by six data files. The first, file SYMBOL, contains the bit maps which make up the source data for the annotation symbol set. Each of the sixty symbols is represented by a 48x48 bit map packed as forty-eight lines of three sixteen-bit words each. Bit packing is left to right within each word. Once established on the system, this file is read in automatically when Module #7, ANNOTE, is invoked. The annotation symbols are then generated as needed by resampling these bit maps to the user-requested size.

Two 'toy' size images - files MTEST and GIRL6 - are supplied with the package. These images are designed for rapid testing of the modules and to aid with inexpensive exploration of the space of user options. A sample set of pre-defined user parameters - file MSET - is also provided; it is set up to be used with the MTEST image.

Finally, two 'real world' (video frame size) images - GIRLIMG and BLDGIMG - are included to provide more realistic test examples.

Intermodule communication is provided by a set of twelve standard MAPS files and is designed to be transparent to the user. Dedicated mnemonic file names are used, however, so that both user and systems personnel can easily assess status of the process following interruption or deliberate suspension of a MAPS interactive task.

The user is also aided by a series of three 'planning forms' which display the space of user options schematically. These forms have spaces for entry of user-selected options to assist in pre-planning and documenting a MAPS interactive session. Moreover, the planning forms provide a convenient guide to the location of various user option interactions for the use of systems personnel.

The TransMAPS package is completed by two volumes of formal documentation - The TransMAPS User's Manual and this Maintenance Manual.

2.2 TransMAPS Structure

The previous section provides only a tabular listing of the contents of the TransMAPS package. The key structure of TransMAPS is presented in Figure 2-1. This diagram shows the interrelationships among all of the process modules and data files. A brief descriptor of the contents of each file is given along with the corresponding standard file name. Figure 2-1 is the primary portrayal of the overall organization of TransMAPS.

2.3 Host Computer Environment

TransMAPS is intended to run on DEC PDP-11/45, PDP-11/70, and VAX computer systems. It was targeted specifically at the PDP-11/70 in the Image Processing System at RADC. The modules are all written in DEC's FORTRAN IV-PLUS. They are each designed to run under DEC's RSX-11M Operating System as single task loads without overlays. The file systems are such that they should be compatible with either RMS-11 or FCS-11.

The TransMAPS modules do not make use of the VIRTUAL declaration construct so a 'mapped' memory system should not be necessary. Moreover, the restriction to FORTRAN IV-PLUS is thought to involve only the use of

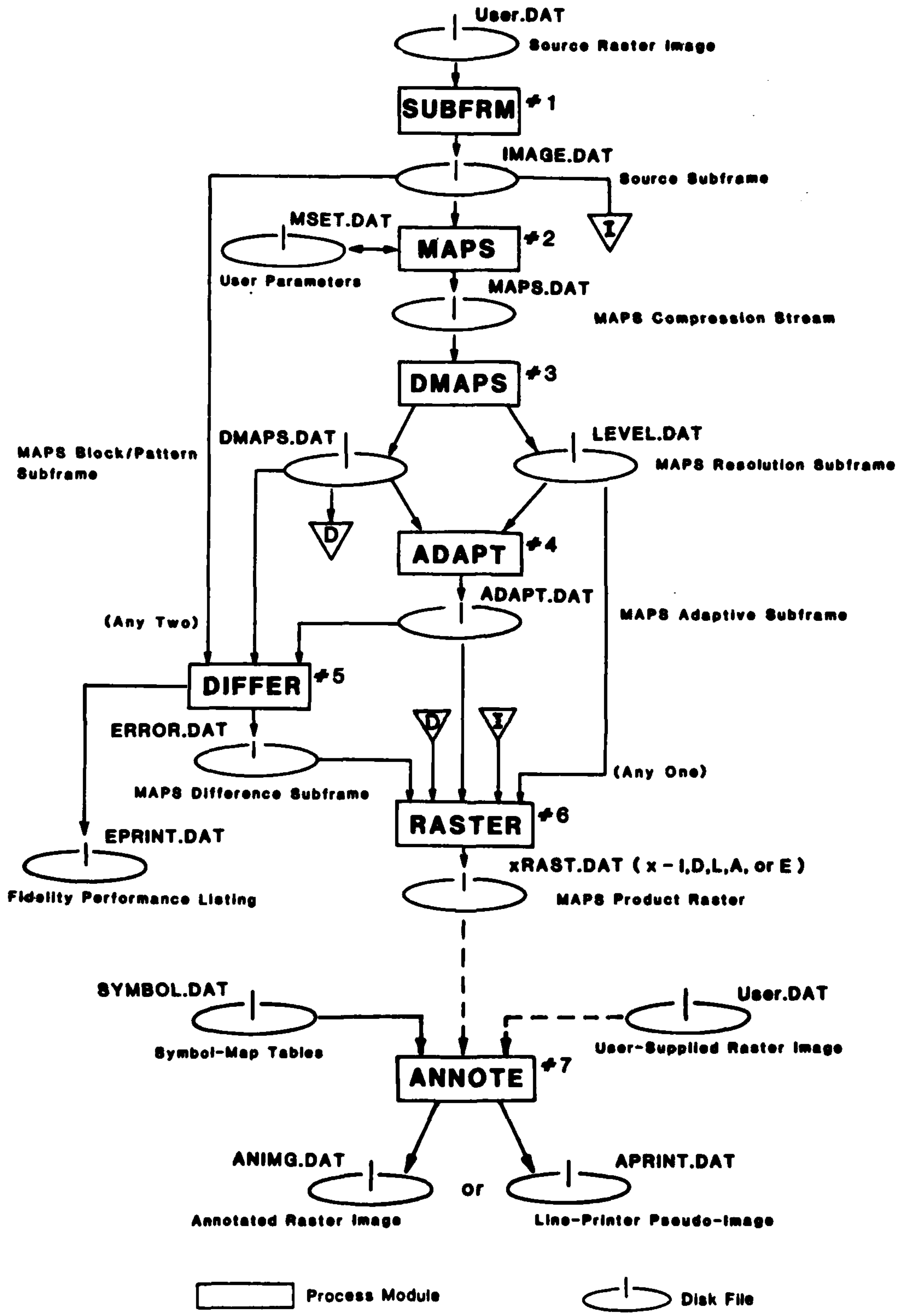


Figure 2-1. TransMAPS Structure

the IISHFT library shift function in Module #7 and the use of Integer*4 arithmetic for a few performance statistics and element count accumulations. At some penalty in execution efficiency, it should thus be possible to modify the code to be compatible with the more restricted DEC FORTRAN compiler if necessary at a particular site.

The user interaction portion of the code tacitly assumes a high-speed channel to the interactive terminal. Thus, some of the query prompts update rather extended text with each user response to provide complete and immediate feedback. If much slower terminal channels (e.g. low-speed telephone modem) or hardcopy terminals are to be employed, these sections of the interaction can be slightly modified to reduce the text redundancy. Guidelines for such changes are provided in Section Five.

SECTION THREE

TRANSMAPS FILES

This section discusses user image file conventions, the formats of standard MAPS files, and the contents of the MAPS file header.

3.1 User File Communication

DEC FORTRAN does not support direct magnetic tape access in any convenient way. Thus, user-supplied source imagery must be entered as a disk file for communication to TransMAPS. Furthermore, there does not appear to be any standard format for image data files which extends across the systems on which TransMAPS is to be used. Thus, the file format for TransMAPS input was chosen to minimize the buffer overhead incurred in the program modules.

TransMAPS assumes that user raster source images will be supplied as sequential binary files written using the SEGMENTED record type. This form allows the physical record size (and consequently the buffer) to be much smaller than the largest logical record size expected. Moreover, the physical records are sufficiently small that file transfer utilities such as FLX can be employed. Because the types of source formatting for a particular site or application are unknown, responsibility for provision of the conversion utilities to establish the SEGMENTED source raster files is presumed to reside with each site. For test purposes, the imagery delivered as part of the TransMAPS package has already been placed in this form. Thus initial installation and checkout can proceed independently from the generation of this image conversion capability. TransMAPS does accommodate both eight bit per pixel and six bit per pixel source forms. For the six bit sources, the so-called DICOMED format convention is assumed; namely, six-bit pixels right justified in eight-bit fields. This convention avoids the 'byte-swap' complications

encountered between DEC (lowest byte first) and many other computer systems (most significant byte first). This six bit convention is also the standard for such imagery at RADC.

3.2 MAPS File Formats

File formats for the remaining MAPS files are determined by access type and file content. Printer listing files are, of course, of FORMATTED type. All raster image files (input or output) are of SEGMENTED binary type. Subframe-organized image files are of FIXED record length binary type suitable for DIRECT access under either FCS-11 or RMS-11 file systems. The MAPS compressed image stream is also written as a FIXED record length binary file with records of 512 bytes. Finally, files containing user parameter sets and the symbol table bit maps are expressed as SEGMENTED binary files.

All MAPS data files with their mnemonic file names, content descriptors, and file type characteristics are listed in Table 3-I.

3.3 Standard MAPS File Header

The various 'internal' MAPS image files - IMAGE, MAPS, DMAPS, LEVEL, ADAPT, and ERROR - all require certain ancillary information in their roles of intermodule communication. This information is carried in a standard MAPS file header which is the first record in each file. In each case, the record length is adjusted to match the corresponding FIXED length for the file. Thus, the header content must fit within the smallest allowed value of this record length - namely, 64 bytes.

TABLE 3-I. TRANSMAPS FILES.

File Characteristics:

Filename	Content	Filetype
User	Source Image, Raster	SEGMENTED
IMAGE	Source Image, Subframes	FIXED *
MSET	MAPS User Parameter Set	SEGMENTED
MAPS	MAPS Compression Stream	FIXED **
DMAPS	MAPS Block/Pattern, Subframes	FIXED *
LEVEL	MAPS Resolution Image, Subframes	FIXED *
ADAPT	MAPS Adaptively Smoothed, Subframes	FIXED *
ERROR	MAPS Difference Image, Subframes	FIXED *
EPRINT	Fidelity Performance Listing	FORMATTED
IRAST	Source Image, Raster	SEGMENTED
DRAST	MAPS Decompressed Image, Raster	SEGMENTED
LRAST	MAPS Resolution Image, Raster	SEGMENTED
ARAST	MAPS Adaptively Smoothed, Raster	SEGMENTED
ERAST	MAPS Difference Image, Raster	SEGMENTED
SYMBOL	Annotation Symbol-Map Tables	SEGMENTED
User	Embedded Input Image, Raster	SEGMENTED
ANIMG	Annotated Image, Raster	SEGMENTED
APRINT	Annotated Pseudo-Image Listing	FORMATTED

* 'FIXED' Subframe Records are 64, 256, or 1024 Bytes
** 'FIXED' Compression Stream Records are 512 Bytes

Currently, the header information occupies the first fifty of these locations and is organized as follows:

```
Standard MAPS File Header:
-------- ---- ---- -------

                                                            Bytes
              Identification:                               -----
              ---------------
                     File Type  0     IMAGE                   2
                                1     MAPS
                                2     DMAPS
                                3     LEVEL
                                4     ADAPT
                                5     ERROR I-D
                                6     ERROR I-A
                                7     ERROR D-A
                     Image Name                               8

              Image Size:
              ----- -----
                     Lines/Image                              2
                     Pixels/Line                              2
                     Bits/Pixel                               2

              Image Partition:
              ----- ----------
                     Subframe Size (8 16 32)                  2
                     Subframe Grid (0=Square, 1=Stagger)      2
                     Subframe Count                           4

              MAPS Results:
              ---- --------
                     MAPSel Count                             4
                     Block/Pattern Vector
                       (1 Bit/Level, Packed Right to Left)    2
                     Optimal Pattern Biases
                       (Level [>0], Low/High)        5x2x2 = 20

              Future Use:
              ------ ----                                    14

                                                             ----
                                                       TOTAL  64
```

Thus, there are fourteen additional bytes available for future expansion before a more elaborate header structure must be sought. Note that a separate record is used for the header on the MAPS-compressed data stream. This, however, is not really necessary since the division into 512-byte records is arbitrary and asynchronous. The MAPS file header could be incorporated into the first fifty bytes of the first record followed immediately by the start of the MAPSel data stream. Only slight modification to the initialization procedures would be required to accomplish this. It can be done in situations where minimum compressed data size is sought.

As might be expected, almost all information in the header is created in the first two modules - SUBFRM and MAPS. The first two bytes encode the type of data in the file. This preserves the integrity of the file contents through subsequent file name changes. The next twenty-two bytes describe the source image and its partition into subframes in program SUBFRM. The remaining twenty-six bytes characterize the MAPS compression and the optimum pattern bias values needed for decompression. This data is generated in Module #2, MAPS. The unused header bytes might be employed in the future to carry the control selections used to create alternate product images.

SECTION FOUR

INSTALLATION

This section describes the procedures required to transfer TransMAPS from its source computer-compatible magnetic tape to a functional system.

4.1 FLX File Transport

The TransMAPS software and support data resides on nine-track magnetic tape (density 800 bpi) in the form of twenty FLX files. These are further subdivided into three file categories - seven FORTRAN source code files, seven FORTRAN object code files, and six data files.

The command sequence to enter these files on the system is as follows:

ALL MMu:	Allocate tape drive
FLX MMu:[50,27]/LI	List FLX tape contents (optional)
FLX SY:[g,m]/RS = MMu:[50,27]*.*/DO	Transfer all TransMAPS files
PIP/LI	List transferred files (optional)
DEA MMu:	Deallocate tape drive

Here, 'u' is the physical unit number of the tape drive on which the TransMAPS FLX tape is mounted and [g,m] is the UIC (user identification code) under which the TransMAPS software is to be installed. The UIC under which the system was originally transferred to tape was [50,27].

The file names and sizes which should appear in the directory listings are as follows:

File	Content	FLX size
SF.FSV	SUBFRM FORTRAN Source Code	29
MP.FSV	MAPS FORTRAN Source Code	67
DM.FSV	DMAPS FORTRAN Source Code	20
AD.FSV	ADAPT FORTRAN Source Code	43
DF.FSV	DIFFER FORTRAN Source Code	28
RS.FSV	RASTER FORTRAN Source Code	23
AI.FSV	ANNOTE FORTRAN Source Code	61
SF.OBJ	SUBFRM Object Code	26
MP.OBJ	MAPS Object Code	71
DM.OBJ	DMAPS Object Code	30
AD.OBJ	ADAPT Object Code	38
DF.OBJ	DIFFER Object Code	25
RS.OBJ	RASTER Object Code	17
AI.OBJ	ANNOTE Object Code	67
SYMBOL.BIN	Symbol Map Tables	37
GIRL6.BIN	Girl 6-bit 'Toy' Image	34
MTEST.BIN	MAPS Logic Test 'Toy' Image	46
MSET.BIN	Sample User Parameter Set	2
BLDGIMG.BIN	'Building Scene' Video Image	631
GIRLIMG.BIN	'IEEE Girl' Video Image	631

4.2 Data Files

The six TransMAPS data files are all transferred under file type '.BIN'. Each of these files is in SEGMENTED binary format. The choice of the '.BIN' type allows FLX to default automatically to Formatted Binary mode. Note that in the FLX tape versions of the files, binary headers and checksums are added to the data. This means that the FLX file size for such files will typically be somewhat larger than the corresponding Files-11 versions. These differences will be reflected in the file sizes reported when PIP is used to list the directory after file transfer. In the Files-11 disk environment, for example, the 'video' image sizes are each 608 blocks.

4.3 FORTRAN Source Code Files

The FORTRAN source code for the seven TransMAPS modules was transferred with an arbitrary file type of '.FSV'. This was done to permit retention of the original version of the source code if later versions are created and then all but the most recent eliminated by a PIP purge switch.

4.4 FORTRAN Object Code Files

The object code for the seven TransMAPS modules was produced by the FORTRAN IV-PLUS compiler (F4P) under RSX-11M on a PDP-11/70. Here, the standard file type, '.OBJ' was used for the transfer. This file type is another of the small class of names for which FLX correctly defaults to the Formatted Binary mode.

4.5 File Renaming and Protection

Note that for both the source and object code files, the file names have been shortened to two-character mnemonics. Experienced users will find these short names useful in reducing the keystrokes needed to invoke the processes during interactive execution. Inexperienced users, however, will probably prefer the security of the longer mnemonics. Thus, it is suggested that the '/EN' switch be used with PIP to enter both the full and short versions of the file name as synonyms in the file directory. This probably need be done only for the task files, type '.TSK', after they have been generated.

The program modules will expect the data files to exhibit type '.DAT'. Thus, the '.BIN' files should all be renamed using the PIP switch '/RE'. Alternatively, the '.BIN' files can be retained as backup and copies created by PIP with a file type of '.DAT'.

In addition to or as an alternative strategy to protecting the key TransMAPS files by using multiple copies with different names, the explicit protection status of the files can be changed via the PIP switch '/PR' with appropriate subswitches. The source code files and the critical data files (SYMBOL.DAT, MTEST.DAT, and GIRL6.DAT) can all be changed to read-only through the switch/subswitch sequence '/PR/SY:R/OW:R/GR:R/WO:R'. The sample user parameter set file, MSET.DAT, can be given deletion protection by replacing the ':R' entries with ':RWE'. Read-only protection can also be applied to the '.TSK' task files once they have been built into the system. Such protection redefinition should help to insure that the functional package is not inadvertently deleted during file maintenance operations where 'wild cards' are sometimes used in some of the file specification fields.

4.6 Program Compilation

In principle, task building could proceed directly from the set of TransMAPS object code files provided as part of the system. At new sites, however, it is probably safer to recompile the FORTRAN source code. This is done for each of the seven modules by the command line:

```
F4P File.OBJ,File.LST=File.FSV   or   F4P File,File=File.FSV.
```

Here, 'File' takes on the 'values' SF, MP, DM, AD, DF, RS, and AI. Note that this compilation string produces a source code listing with a cross-reference map as well as the object code file for each module.

4.7 Task Building

Following compilation, each module must be 'task built' into a loadable form. Task building of most of the TransMAPS modules also requires exercise of some of the 'TKB' options to change various default settings.

The general sequence and the specific options are outlined in the following tabulation:

Task Build Options:

General Sequence:

```
>TKB (cr)
TKB> task/FP,map=object (cr)
TKB> / (cr)
ENTER OPTIONS:
TKB> option1=n1 (cr)
TKB> option2=n2 (cr)
TKB> // (cr)
```

Program Module	Options Required
SUBFRM (SF)	MAXBUF=1024
MAPS (MP)	MAXBUF=1024
DMAPS (DM)	MAXBUF=1024
ADAPT (AD)	MAXBUF=1024
DIFFER (DF)	MAXBUF=1024 ACTFIL=5
RASTER (RS)	MAXBUF=1024
ANNOTE (AI)	None

Note that all but one of the options involves increasing the maximum file buffer size to accommodate the largest expected FIXED record length in the subframe-organized files. The ACTFIL increase in the DIFFER module (from the default value of four) is required by the addition of the performance evaluation listing, EPRINT.DAT, to the normal set of image data files.

It might also be noted that a shorter task build command can be used with the ANNOTE module since no TKB options are required there. The appropriate command line takes the following form:

TKB AI.TSK/FP,AI.MAP=AI.OBJ or TKB AI/FP,AI=AI.

The '/FP' switch which shows up on the task file in all cases here is used to invoke the 'Floating Point Processor'. Under some versions of RSX-11M, this switch is required even if the module contains no instructions involving floating point processes. Failure to include this switch is not detected at task build but generates run time error #2: "TASK INITIALIZATION FAILURE".

4.8 Test and Verification

Once the seven modules have undergone task building and the data files have been renamed (or copied) to type '.DAT', the TransMAPS software should be ready for application. Several quick tests can be run to verify the system. A suggested sequence to check the interface modules proceeds as follows:

- RUN SF on GIRL6.DAT to convert the 120 line by 128 pixel by 6 bit raster image to 8 bit subframe form;
- RUN RS on IMAGE.DAT to convert the resultant subframe image back to raster organization; and
- RUN AI on IRAST.DAT with annotation and in the pseudo-image mode to replicate the output shown in Figure 4-1.

This process will verify operation of these three modules and will exercise the SYMBOL.DAT bit map tables. It also provides immediate visual confirmation via the line printer even on systems where no other image display is available. Note that the printer should be switched to an eight-line-per-inch mode to more closely approximate square pixels (an 8:10 ratio rather than 6:10).

The IMAGE.DAT file for the girl 'toy' image can also be used to test the various processes directly related to MAPS. Here it is suggested that

some or all of the examples in Section Nine of the User's Manual be reproduced. Because of the small image size, such runs go very quickly and provide immediate feedback via the printer.

Finally, the detailed MAPS logic can be verified with the diagnostic image MTEST.DAT as displayed in Figure 4-2. Examples of the use of this image are presented in Section Eight of the User's Manual. The sample user parameter set on file MSET.DAT can be used in conjunction with this image to check and demonstrate the macro-fidelity control capabilities of TransMAPS. The user options can be reviewed and updated to implement various combinations of the following diagnostic tests:

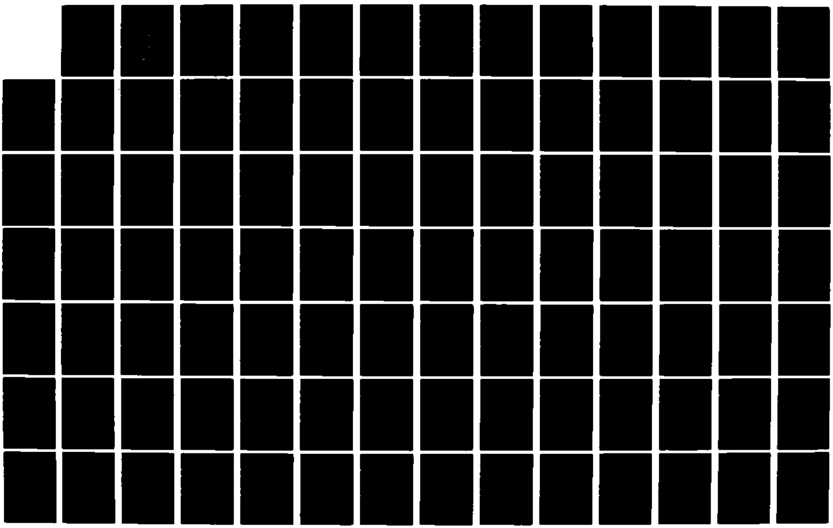
```
Compression-Logic Diagnostic Tests:
---------------- ---------- ------

        Image Partition and Macro-Fidelity Control:
        ----- --------- --- -------------- -------
                Input Image Line and Pixel Skips
                Subframe Phasing (Square and Staggered)
                Macro-Partition Group Assignment

        Micro-Fidelity Control:
        -------------- -------
                Zigzag Sequencing
                Contrast Control as a Function of Transition Level
                Contrast Control as a Function of Contrast Type
                Pattern Code Assignment

        Gray-Scale Manipulation:
        ---------- ------------
                Contrast Space Quad Sort
                Intensity Space Quad Sort
                Intensity Reset Assignment
```

Successful completion of such checks should then insure an operational MAPS compression capability suitable for further exploration or functional image data base preparation.

Figure 4-1. MAPS Familiarization Test Image

AD-A120 982 TRANSPORTABLE MAPS SOFTWARE VOLUME I(U) CONTROL DATA CORP MINNEAPOLIS MN INFORMATION SCIENCES DIV A E LABONTE JUL 82 9199500 RADC-TR-82-194 3/4

UNCLASSIFIED F30602-80-C-0326 F/G 20/6 NL



1.0 4.5 5.0 5.6 6.3 7.1 8.0 9.0 2.8 3.2 3.6 4.0 2.5 2.2 2.0

1.1 1.8

1.25 1.4 1.6

MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

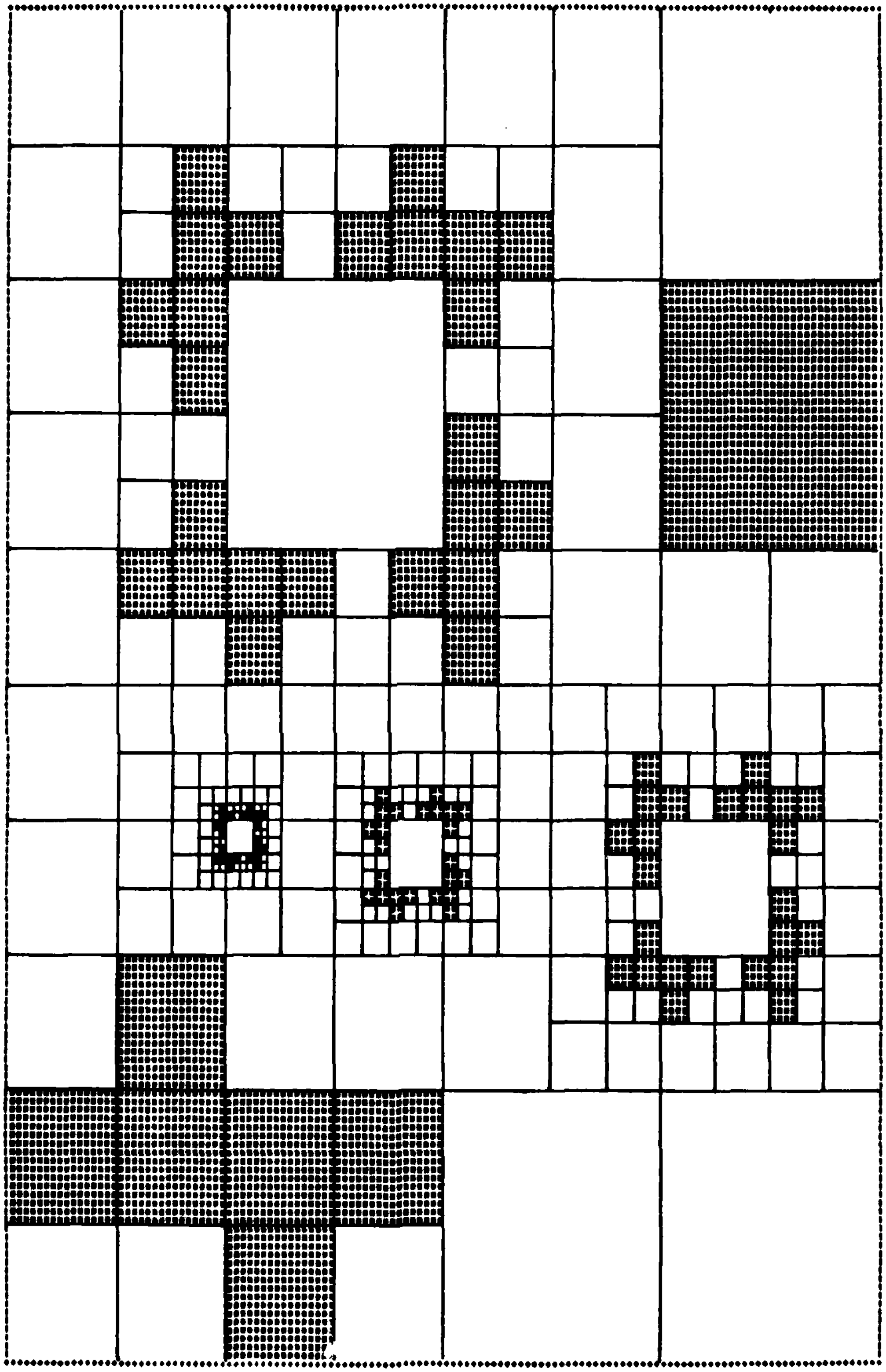


Figure 4-2. MAPS Compression-Logic Diagnostic Image

SECTION FIVE

MODIFICATION

This section discusses three known circumstances where TransMAPS modifications might be desired - default parameter changes, accommodation of interactive terminal limitations, and system-specific code constraints.

5.1 Default Settings

TransMAPS contains an extensive space of user options and each option generally has some preset default selection. As experience with the package grows or as typical image characteristics emerge at a particular site, it may prove advantageous to change some of these default parameter settings. For example, it may be found that a particular image frame size is encountered on a regular basis. In this instance, the default line count and pixel count for the source imagery might be set to this size. As another example, a different standard subframe partition may be best for a large class of images and the corresponding controls can be selected for the default. Or again, a different set of contrast control generator parameters (taper, step fraction, or step bias) may be desired for 'normal' operation.

In order to support such changes, the default parameters for each module have been collected and placed together near the beginning of the user interaction subroutine for that program. Moreover, to make these default settings easy to locate, the relevant program lines have been 'flagged' with on-line COMMENTs of the form '! Default'. In order to effect a default change, the system personnel need merely isolate the appropriate setting in the source code, modify its value, recompile the module, and rebuild the resultant task.

5.2 Hard-copy or Low-baud-rate Terminals

TransMAPS has been implemented to give extensive feedback in the user interaction by providing immediate update and display of user option selections. In addition, extended prompts have been used to remind the user of the range of choices available at various points. This strategy is very helpful in a system with fast communication channels and video rate terminal displays. However, slow channels such as low-speed phone lines or hard-copy terminals such as tele-type or DEC-writer are less satisfactory for this approach because of their long response time.

To reduce user frustration in such environments, it is probably desirable to restructure the interaction to provide updates or complete prompts less often. Reduced frequency of output is recommended in the following three areas:

- In MAPS - Present the updated image macro-fidelity partition matrix only after the user has signalled that all desired line-editing is complete;
- In MAPS - Present the updated micro-fidelity control parameters and corresponding threshold matrix only after a user signal that editing is complete; and
- In ANNOTE - Present the allowed character set only once before the definition of the first annotation message, not as a prompt to every message text definition.

Other user communications which could be shortened are those in ANNOTE which define the message orientation and the message length. Note that in the slowest interaction - the hard-copy terminal - the prior copy itself is available for review. Thus, reissue of prompts is required much less than in a faster but more volatile soft-copy environment.

5.3 System-Specific Code

Two known constructs specific to FORTRAN IV-PLUS were used in TransMAPS. If it is necessary to transport the package to a system where only the FORTRAN compiler is supported, these constructs would have to be simulated by less efficient code.

The first limitation is restricted to Module #7, ANNOTE, where the library shift function, IISHFT, is used in the symbol bit-map resampling processes. The shifts could be replaced by multiplies and divides with powers of two. However, the bit-map tables should probably be redefined as forty-eight lines of six bytes each, rather than as forty-eight lines of three sixteen-bit words each. This is the first step in circumventing the problems of sign interpretation and overflow in the high-order bits of the Integer*2 words. Note that each byte should first be transferred to the lower portion of a normal integer (two bytes) and sign-corrected before any of the arithmetic operations (pseudo-shifts) are carried out. Note also that the 'byte-swap' problem must be accommodated since the bit-maps are stored left to right in the original two-byte words.

The second construct involves use of Integer*4 arithmetic at several points where performance statistics or element counts are accumulated. Since the ordinary FORTRAN compiler supports the Integer*4 data type but only allows Integer*2 arithmetic on such variables, all Integer*4 arithmetic constructs must be converted to compound Integer*2 processes with extensive overflow checking. Should such conversions be needed, an attempt has been made to flag all lines containing Integer*4 arithmetic with on-line COMMENTs of the form '! I*4'. These flags should at least help to localize the conversion process.

SECTION SIX

MAINTENANCE

This section describes common features among the program modules in order to provide a framework for understanding the overall philosophy of implementation used in TransMAPS. This information should provide useful guidance for any necessary software maintenance activities on the package.

6.1 Generic Program Structure

All seven principal TransMAPS modules have essentially the following overall structure:

```
Generic Program Structure:
------- ------- ----------

        User Interaction     -    Subroutine USERx (x = I,M,D,A,E,R)
        File Establishment   -    Subroutine FILESx (x = I,M,D,A,E,R)
        Process Initialization - Subroutine SETUPx (x = I,M,D,A,E,R)
        PRINCIPAL LOOP on Subframes or Lines
```

Each program contains three main preprocess steps - user option interaction, opening and positioning of files, and initialization of tables and variables. The names for the primary subroutines which implement these preprocesses also exhibit a consistent convention - USERx, FILESx, and SETUPx. Here, 'x' is a single-character mnemonic for the host module which calls the routine. Note that the calling order for these preprocesses does vary among the modules since the file header information is needed to direct the user interaction in some cases, while the user interaction selects the relevant files in others. Such usage is clear from context in the actual program listings.

The body of each program is then typically a loop which iterates on lines or subframes within the image(s). Both the preprocesses and the body of each program are further modularized at points of natural process division. This partitions the implementation into 'graspable' chunks with an average routine length of less than fifty lines including continuations but excluding COMMENTs. Both the number of subroutines and the overall task length for each module are listed in the following tabulation:

Module Characteristics:

Program	Subroutines	Task Length
SUBFRM	6	31936
MAPS	15	31488
DMAPS	6	18720
ADAPT	10	27968
DIFFER	5	25312
RASTER	5	29664
ANNOTE	14	31840

Note that with the exception of Module #3, DMAPS, all of the programs use a large fraction of the available 32K task space. Indeed, three of the modules just fit within the constraint imposed by the sixteen bit word length.

6.2 Integrated User Interaction

In general, the user interaction portions of the code were not further partitioned into smaller chunks. In this area, it was felt that the flow of the interaction was easier to follow if it were not distributed among several small routines.

The sequence of interactions tends to exhibit a natural punctuation. Typically, the definition of each user option or parameter proceeds through a sequence of fairly well-defined steps. First, the current value (initially the default) is checked for consistency with parameters defined previously in the interaction and reset as necessary. Next, the user is prompted with the parameter under consideration, its current value, and its current allowed range (if applicable). This prompt is in the form of a query to which the user can respond with either a new value or 'no change' as desired. The user's response is then used to update the parameter and it is again checked and corrected for consistency with the allowed range. The interaction then proceeds to the next option or parameter. Thus, the code comes naturally in a sequence of easily grasped packets.

In the TransMAPS implementation of the user interactions, the packets are further set out by indenting the consistency checking operations. Moreover, the FORMAT statements containing the query communications are placed at the location of the corresponding packets (rather than being collected at the beginning or end of the routine). These FORMAT statements then serve as integral documentation of the interaction and the flow of code need not be interrupted with separate COMMENTs.

6.3 Subroutine Communication

Modularization in MAPS is intended more to give conceptual organization

to the process than to isolate multiple-use segments of code. The flow is characterized by a sequence of processes on a common body of data using common control parameters. Thus, formal parameters to accommodate varying points of application are not needed for subroutine transfer. The data and control parameters are then conveniently organized into named COMMON blocks for interroutine communication.

These labeled COMMON blocks serve to give further structure to the process and to provide mnemonic groupings in the data and parameter spaces. Indeed, such blocking provides the organizing principle for construction of an effective DATA DICTIONARY for each module. This information is then documented by including it in the program COMMENTs.

6.4 Intra-Program Documentation

For a capable programmer, the flow of code itself is the key documentation in a program and extensive in-line COMMENTs often prove distracting. Hence, a consistent pattern of header COMMENT blocks has been adopted for the internal documentation of the TransMAPS modules. An extended block is given at the beginning of each of the seven main modules with abbreviated headers given in each of the other routines. The generic formats for these headers are summarized in Table 6-I.

Note that the main block contains a brief process descriptor, file communication definitions, a user interaction outline, a 'structured' process hierarchy, and a data dictionary. The subroutine headers contain a brief description of purpose plus the CALLing links to and from the routine.

The remaining seven sections of this manual present the detailed listings of this annotated TransMAPS source code.

Table 6-I. GENERIC MAPS PROGRAM COMMENTS

```
COMMENT Formats:
------- --------

    Principal Module Headers:
    --------- ------ --------
        C
        C  +-----------------------------------------+
        C  |                                         |
        C  |  TransMAPS Module #n:  process descriptor |
        C  |                                         |
        C  +-----------------------------------------+
        C
        C                      Control Data Corporation - 1982
        C
        C Files: Unit  Name  Content            From/To  Type
        C ------
        C  In/Out  n   zzzzz descriptor         module   SEGMENTED
        C                                             or DIRECT
        C                                             or FIXED SEQ.
        C                                             or FORMATTED
        C
        C User Interaction:
        C -----------------
        C       principal interactive parameter groups
        C
        C Program Structure:
        C ------------------
        C       subroutine calling hierarchy with brief process outline
        C
        C COMMON Block Communication:
        C ---------------------------
        C       /blockname/     descriptor             length (1*2 words)
        C                         host routine names
        C
        C           variable    [datatype]      descriptor
        C
        C               (These lists provide a module DATA DICTIONARY)
        C
        C order conventions or geometry definitions (if appropriate):
        C -----------------------------------------------------------

    Subroutine Headers:
    ---------- --------
        C
        C  Purpose:  brief process description
        C
        C  CALLed from:  calling routine name(s)
        C
        C  CALLs:  called routine name(s)
        C
        C  geometry definitions if appropriate:
        C-------------------------------------------------------------

    On-line Flags:
    ------- ------

        Expression      Comment              Function
        ----------      -------              --------
            ...        ! Default       (default values set in USERx)
            ...        ! I*4           (four-byte integer arithmetic)
```

SECTION SEVEN

TRANSMAPS MODULE #1: RASTER TO SUBFRAME CONVERSION

7.1 Program Characteristics

Program Names: SUBFRM or SF

Subroutines:	USERI	
	FILESI	
	SETUPI	
	SQUARE	
	STAGGR	
	LINEIN	
Files:	User.DAT	(input user raster)
	IMAGE.DAT	(output)

Task Build Options: MAXBUF = 1024

Task Size: 31936

7.2 Source Listing

The COMMENT-annotated source listing for SUBFRM follows:

```
      PROGRAM SUBFRM
C
C  +----------------------------------------------------------+
C  |                                                          |
C  |  TransMAPS Module #1:  Raster to Subframe Conversion     |
C  |                                                          |
C  +----------------------------------------------------------+
C
C                                       Control Data Corporation - 1982
C
C Files: Unit  Name    Content                         From/To  Type
C ------
C   In    2    User    Source raster image              User    SEGMENTED
C   Out   3    IMAGE   Source image, subframes          MAPS    DIRECT
C                                                    or DIFFER
C                                                    or RASTER
C
C User Interaction:  In Subroutine USERI
C -----------------
C       Source Image Identification
C       Source Image Position Specification
C       Source Image Size Specification
C       Source Image Partition Specification
C
C Program Structure:
C ------------------
C       PROGRAM SUBFRM
C           CALL USERI   Specify image name, position, size, partition
C           CALL FILESI   Open and position input file, open output file
C           CALL SETUPI   Characterize image partition and padding
C           IF square grid:
C               CALL SQUARE   Convert raster: 8x8 16x16 32x32 subframes
C                   Loop on rows of subframes in line direction
C                       Loop on 8-line swathes within subframe rows
C                           CALL LINEIN   8 calls, line input, 6-8 bit
C                           Loop on subframes in pixel direction
C                               Input prior partial subframe (if 16 32)
C                               Update subframe with 8 line segments
C                               Output updated subframe
C        or IF staggered grid:
C               CALL STAGGR   Convert raster: 8x8 staggered subframes
C                   Loop on lines
C                       CALL LINEIN   Input next line, 6-8 bit conversion
C                       Loop on subframes completed on this line
C                           Extract subframe from recirculating buffer
C                           Output subframe
C
C COMMON Block Communication:
C ---------------------------
C    .Blank.    Raster image input data              Length: 16128
C                   SUBFRM,SQUARE,STAGGR,LINEIN
C
C       IBUF(4032,8)    [Byte]  Block or recirculating 8-line buffer
C
```

```
C     /HEADER/    Standard MAPS file header                 Length: 32
C                      SUBFRM,USERI,FILESI,SETUPI
C
C        IFILE             [I*2]    File type
C        INAME(8)          [Byte]   User-selected image name
C        NL                [I*2]    Number of lines in source image
C        NP                [I*2]    Number of pixels in source image
C        NB                [I*2]    Number of bits/pixel in source image
C        KSF               [I*2]    Kind of subframe 8x8 16x16 32x32
C        IGRD              [I*2]    Subframe grid: square(0)/staggered(1)
C        NS                [I*4]    Total subframe count
C        MC                [I*4]    MAPSel count
C        MIXBP             [I*2]    Packed block(0)/pattern(1) mode (rt-lft)
C        IBV(5,2)          [I*2]    Optimal pattern biases by level,low/high
C        IPAD(7)           [I*2]    Space for future extension
C
C     /IMAGIN/    Source file and position data             Length: 7
C                      USERI,FILESI,SETUPI
C
C        FILNAM(10)        [Byte]   Source raster file name
C        LSKP              [I*2]    Lines to skip into source raster
C        KSKP              [I*2]    Pixels to skip into source raster
C
C     /LINEUP/    Line input control parameters             Length: 7
C                      SETUPI,SQUARE,STAGGR,LINEIN
C
C        IP                [I*2]    Initial pixel to retain (KSKP+1)
C        IPR               [I*2]    Total input pixels to read
C        LP                [I*2]    Last pixel with subframe completion pad
C        NBT               [I*2]    Bit count flag 6(1) or 8(0)
C        L8                [I*2]    Current line mod 8
C        LK                [I*2]    Current line in input raster incl skips
C        LSKPT             [I*2]    Lines skipped (copy of LSKP)
C
C     /SFDATA/    Subframe output data                      Length: 512
C                      SETUPI,SQUARE,STAGGR
C
C        ISF(1024)         [Byte]   Subframe image assembly array
C
C     /SFTEMP/    Subframe and image size parameters        Length: 6
C                      SETUPI,SQUARE,STAGGR,LINEIN
C
C        NLT               [I*2]    Number of lines in image
C        NPT               [I*2]    Number of pixels in image with pad
C        KSFT              [I*2]    Subframe size: edge
C        KSQT              [I*2]    Subframe size: pixel count
C        NST               [I*4]    Subframe count
C
C     /SQR/       Square grid partition controls            Length: 3
C                      SETUPI,SQUARE
C
C        NPS               [I*2]    Number of pixel-direction subframes
C        NLS               [I*2]    Number of line-direction subframes
C        NSWTH             [I*2]    Number of swathes/subframe (1 2 4)
C
```

```
C     /STGR/      Staggered grid partition controls       Length: 24
C                     SETUPI,STAGGR
C
C         NPSL(8)          [I*2]   Number of staggered subframes in pixel
C                                    direction for each startline mod 8
C         NLSL(8)          [I*2]   Number of staggered subframes in line
C                                    direction for each startline mod 8
C         IPSL(8)          [I*2]   Initial pixel of first staggered
C                                    subframe for each startline mod 8
C
C----------------------------------------------------------------------
        COMMON IBUF(4032,8)
        BYTE IBUF
        COMMON /HEADER/ IFILE,INAME(8),NL,NP,NB,KSF,IGRD,NS,MC,MIXBP,
     +    IBV(5,2),IPAD(7)
        BYTE INAME
        INTEGER*4 NS,MC
        CALL USERI
        CALL FILESI
        CALL SETUPI
        TYPE 500,INAME,NS
500     FORMAT(/,1X,'CONVERTING IMAGE ',8A1,' TO',I7,' SUBFRAMES')
        IF(IGRD.NE.0) GO TO 110
        CALL SQUARE
        GO TO 120
110     CALL STAGGR
120     CONTINUE
        CLOSE(UNIT=2)
        CLOSE(UNIT=3)
        END
```

```
      SUBROUTINE USERI
C
C  Purpose: User interaction for raster to subframe conversion
C               Source image identification
C               Source image position
C               Source image size
C               Source image partition
C
C  CALLed from: SUBFRM
C
C-----------------------------------------------------------------------
      COMMON /HEADER/ IFILE,INAME(8),NL,NP,NB,KSF,IGRD,NS,MC,MIXBP,
     +  IBV(5,2),IPAD(7)
      BYTE INAME
      INTEGER*4 NS,MC
      COMMON /IMAGIN/ FILNAM(10),LSKP,KSKP
      BYTE FILNAM
      DIMENSION NAMET(10)
      BYTE NAMET,NAME1
      EQUIVALENCE (NAME1,NAMET(1))
      DATA IFILE/0/                                       ! Default
      DATA INAME/8*1H /                                   ! Default
      DATA NL,NP,NB,KSF,IGRD/480,624,8,8,0/               ! Default
      DATA NS,MC,MIXBP,IBV,IPAD/2*0,18*0/
      DATA FILNAM/1HF,1HO,1HR,1HO,1HO,1H2,4*0/            ! Default
      DATA LSKP,KSKP/2*0/                                 ! Default
      DATA MPIX/4000/
      TYPE 500
500   FORMAT(/,1X,45(1H*),/,1X,'*',43X,'*',/,1X,
     +  '* MAPS RASTER TO SUBFRAME CONVERSION MODULE *',
     +  /,1X,'*',43X,'*',/,1X,45(1H*))
100   CONTINUE
      TYPE 510
510   FORMAT(/,3X,'SOURCE IDENTIFICATION:')
      TYPE 520,FILNAM
520   FORMAT(/,5X,'SOURCE RASTER FILENAME? (UP TO 9 CHARACTERS)  ',
     +  10A1)
      ACCEPT 1,NAMET
1     FORMAT(10A1)
      IF(NAME1.EQ.1H ) GO TO 130
      IF(NAME1.EQ.1H/) GO TO 130
      IF((NAME1.GE.1HA).AND.(NAME1.LE.1HZ)) GO TO 110
      TYPE 530
530   FORMAT(/,1X,'*** FILENAME MUST START WITH LETTER')
      GO TO 100
110   DO 120 I=1,9
      FILNAM(I)=NAMET(I)
      IF(NAMET(I).LE.1H ) FILNAM(I)=0
      IF(NAMET(I).EQ.1H/) FILNAM(I)=0
      IF(I.EQ.1) GO TO 120
      IF(FILNAM(I-1).EQ.0) FILNAM(I)=0
120   CONTINUE
```

```
130     TYPE 540,INAME
540     FORMAT(/,5X,'USER IMAGE NAME? (UP TO 8 CHARACTERS)  ',8A1)
        ACCEPT 1,(NAMET(I),I=1,8)
        DO 140 I=1,8
        IF(NAMET(I).NE.1H ) GO TO 150
140     CONTINUE
        GO TO 170
150     IF(NAME1.EQ.1H/) GO TO 170
        DO 160 I=1,8
        INAME(I)=NAMET(I)
        IF(INAME(I).LE.1H ) INAME(I)=1H
160     CONTINUE
170     TYPE 550
550     FORMAT(/,3X,'SOURCE IMAGE POSITION:')
        TYPE 560,LSKP
560     FORMAT(/,5X,'NUMBER OF LINES TO SKIP?',I5,5X,'(/ = NO CHNG)')
        ACCEPT *,LSKP
        TYPE 570,MPIX,KSKP
570     FORMAT(5X,'NUMBER OF PIXELS TO SKIP? (<',I5,')',I5,4X,
     +    '(/ = NO CHNG)')
        ACCEPT *,KSKP
        IF(KSKP.GE.MPIX) KSKP=MPIX-1
        TYPE 580
580     FORMAT(/,3X,'SOURCE IMAGE SIZE:')
        TYPE 590,NL
590     FORMAT(/,5X,'NUMBER OF LINES TO PROCESS?',I5,'  (/ = NO CHNG)')
        ACCEPT *,NL
        MXP=MPIX-KSKP
        IF(NP.GT.MXP) NP=MXP
        TYPE 600,MXP,NP
600     FORMAT(5X,'NUMBER OF PIXELS TO PROCESS? (UP TO',I5,')',I5,
     +    '   (/ = NO CHNG)')
        ACCEPT *,NP
        IF(NP.GT.MXP) NP=MXP
        TYPE 610,NB
610     FORMAT(5X,'NUMBER OF BITS/PIXEL? (6 8)',I3,4X,'(/ = NO CHNG)')
        IT=NB
        ACCEPT *,IT
        IF((IT.EQ.6).OR.(IT.EQ.8)) NB=IT
        TYPE 620
620     FORMAT(/,3X,'SOURCE IMAGE PARTITION:')
        TYPE 630,KSF
630     FORMAT(/,5X,'SUBFRAME EDGE? (8 16 32)',I3,7X,'(/ = NO CHNG)')
        IT=KSF
        ACCEPT *,IT
        IF((IT.EQ.8).OR.(IT.EQ.16).OR.(IT.EQ.32)) KSF=IT
        IF(KSF.NE.8) GO TO 180
```

```
      LGRD=1HN
      IF(IGRD.NE.0) LGRD=1HY
      TYPE 640,LGRD
640   FORMAT(5X,'STAGGER GRID? (Y OR N) ',A1)
      ACCEPT 1,LIT
      IF(LIT.EQ.1HN) IGRD=0
      IF(LIT.EQ.1HY) IGRD=1
      GO TO 190
180   IGRD=0
190   TYPE 650
650   FORMAT(//,3X,'USER SPECIFICATION COMPLETE:',/,3X,28(1H*),//,5X,
     +  'REVIEW? (Y OR N)  N')
      ACCEPT 1,LIT
      IF(LIT.EQ.1HY) GO TO 100
      RETURN
      END
```

```
      SUBROUTINE FILESI
C
C  Purpose: Open and position files
C
C  CALLed from: SUBFRM
C
C-----------------------------------------------------------------------
      COMMON /HEADER/ IFILE,INAME(8),NL,NP,NB,KSF,IGRD,NS,MC,MIXBP,
     +  IBV(5,2),IPAD(7)
      BYTE INAME
      INTEGER*4 NS,MC
      COMMON /IMAGIN/ FILNAM(10),LSKP,KSKP
      BYTE FILNAM
      OPEN(UNIT=2,TYPE='OLD',NAME=FILNAM,FORM='UNFORMATTED',READONLY)
      IF(LSKP.LE.0) GO TO 130
      DO 110 L=1,LSKP
      READ (2,END=120,ERR=120)
110   CONTINUE
      GO TO 130
120         TYPE 500,L
500         FORMAT(/,1X,'*** EOF/ERR AT SKIP LINE',I5)
            STOP
130   CONTINUE
      LSF=(KSF*KSF)/4
      OPEN(UNIT=3,TYPE='NEW',NAME='IMAGE',FORM='UNFORMATTED',
     +  RECORDTYPE='FIXED',RECORDSIZE=LSF,ACCESS='DIRECT')
      RETURN
      END
```

```
      SUBROUTINE SETUPI
C
C  Purpose: Establish subframe partition parameters
C           Write subframe output file standard MAPS header
C
C  CALLed from: SUBFRM
C
C-----------------------------------------------------------------
      COMMON /HEADER/ IFILE,INAME(8),NL,NP,NB,KSF,IGRD,NS,MC,MIXBP,
     +   IBV(5,2),IPAD(7)
      BYTE INAME
      INTEGER*4 NS,MC
              DIMENSION JHEAD(32)
              EQUIVALENCE (JHEAD(1),IFILE)
      COMMON /IMAGIN/ FILNAM(10),LSKP,KSKP
      BYTE FILNAM
      COMMON /LINEUP/ IP,IPR,LP,NBT,L8,LK,LKSPT
      COMMON /SQR/ NPS,NLS,NSWTH
      COMMON/STGR/ NPSL(8),NLSL(8),IPSL(8)
              DIMENSION JPSL(8),JLSL(8)
      COMMON /SFTEMP/ NLT,NPT,KSFT,KSQT,NST
      INTEGER*4 NST
      COMMON /SFDATA/ ISF(1024)
      BYTE ISF
              DIMENSION JSF(512)
              EQUIVALENCE (JSF(1),ISF(1))
      INTEGER*4 ISTAR4
      DATA IPSL/1,25,49,9,33,57,17,41/
      DATA JPSL/7,4,1,6,3,0,5,2/,JLSL/7,14,13,12,11,10,9,8/
      DATA JSF/512*0/
      NPS=(NP-1)/KSF+1
      IP=KSKP+1
      IPR=KSKP+NP
      LP=KSKP+KSF*NPS
      NBT=0
      IF(NB.EQ.6) NBT=1
      LSKPT=LSKP
      IF(IGRD.NE.0) GO TO 110
C
C     Square Grid Partition
C
      NLS=(NL-1)/KSF+1
      ISTAR4=NPS
      NS=ISTAR4*NLS                                         ! I*4
      NSWTH=1
      IF(KSF.EQ.16) NSWTH=2
      IF(KSF.EQ.32) NSWTH=4
      GO TO 130
110   CONTINUE
```

```
C
C       Staggered Grid Partition
C
        NS=0
        DO 120 J=1,8
        NPSL(J)=(NPS+JPSL(J))/8
        ISTAR4=NPSL(J)
        NLSL(J)=(NL+JLSL(J))/8
        NS=NS+NLSL(J)*ISTAR4                                  ! I*4
120     CONTINUE
130     NLT=NL
        NPT=KSF*NPS
        KSFT=KSF
        KSQT=KSF*KSF
        NST=NS
        DO 140 J=1,32
140     JSF(J)=JHEAD(J)
        WRITE (3'1) (ISF(J),J=1,KSQT)
        RETURN
        END
```

```
      SUBROUTINE SQUARE
C
C  Purpose: Convert source image raster to square grid of subframes
C             Loop on rows of subframes in line direction
C                 Loop on 8-line swathes within subframe rows
C                     Input next 8 lines
C                     Loop on subframes in pixel direction
C                         Input prior partially-completed subframe
C                         Update subframe with 8 line segments
C                         Output updated subframe
C
C  CALLed from: SUBFRM
C
C  CALLs: LINEIN
C
C-----------------------------------------------------------------------
      COMMON IBUF(4032,8)
      BYTE IBUF
      COMMON /LINEUP/ IP,IPR,LP,NBT,L8,LK,LSKPT
      COMMON /SQR/ NPS,NLS,NSWTH
      COMMON /SFTEMP/ NLT,NPT,KSFT,KSQT,NST
      INTEGER*4 NST
      COMMON /SFDATA/ ISF(1024)
      BYTE ISF
      INTEGER*4 KREC,JREC
      L=0
      KREC=1
      DO 300 JLS=1,NLS
          DO 270 JSWTH=1,NSWTH
              DO 140 JLIN=1,8
              L=L+1
              LK=L+LSKPT
              IF(L.LE.NLT) GO TO 120
              JPREV=JLIN-1
              IF(JPREV.EQ.0) JPREV=8
                  DO 110 JPIX=1,NPT
110               IBUF(JPIX,JLIN)=IBUF(JPIX,JPREV)
              GO TO 140
120           L8=JLIN
              CALL LINEIN
140           CONTINUE
              JREC=KREC
              JIP=1
              JLP=KSFT
              ISQT=8*KSFT*(JSWTH-1)
              DO 260 JPS=1,NPS
              JREC=JREC+1                                     !  I*4
              IF(JSWTH.NE.1) GO TO 220
                  DO 210 JSQT=1,KSQT
210               ISF(JSQT)=0
              GO TO 230
220               READ (3'JREC) (ISF(JSQT),JSQT=1,KSQT)
```

```
230           JSQT=ISQT
                 DO 250 JLIN=1,8
                    DO 240 JPIX=JIP,JLP
                    JSQT=JSQT+1
240                 ISF(JSQT)=IBUF(JPIX,JLIN)
250              CONTINUE
                 WRITE (3'JREC) (ISF(JSQT),JSQT=1,KSQT)
              JIP=JIP+KSFT
260           JLP=JLP+KSFT
270        CONTINUE
300     KREC=KREC+NPS                                       !  I*4
        RETURN
        END
```

```
      SUBROUTINE STAGGR
C
C  Purpose: Convert source image raster to staggered grid of subframes
C               Loop on lines
C                   Input next line
C                   Loop on subframes completed on this line
C                       Extract subframe from recirculating buffer
C                       Output subframe
C
C  CALLed from: SUBFRM
C
C  CALLs: LINEIN
C
C-----------------------------------------------------------------------
      COMMON IBUF(4032,8)
      BYTE IBUF
      COMMON /LINEUP/ IP,IPR,LP,NBT,L8,LK,LSKPT
      COMMON /STGR/ NPSL(8),NLSL(8),IPSL(8)
      COMMON /SFTEMP/ NLT,NPT,KSFT,KSOT,NST
      INTEGER*4 NST
      COMMON /SFDATA/ ISF(1024)
      BYTE ISF
      INTEGER*4 JREC
      JREC=1
      NL7=NLT+7
      DO 300 L=1,NL7
      LK=L+LSKPT
      LSTRT=L.AND."7
      LS1=LSTRT+1
      IF(LSTRT.EQ.0) LSTRT=8
      IF(L.LE.NLT) GO TO 120
          JPREV=LSTRT-1
          IF(JPREV.EQ.0) JPREV=8
          DO 110 JPIX=1,NPT
110       IBUF(JPIX,LSTRT)=IBUF(JPIX,JPREV)
      GO TO 140
120   L8=LSTRT
      CALL LINEIN
140   IF(L.GT.1) GO TO 200
              DO 160 JLIN=2,8
                  DO 150 JPIX=1,NPT
150               IBUF(JPIX,JLIN)=IBUF(JPIX,1)
160           CONTINUE
200   CONTINUE
      JPSL=NPSL(LS1)
      IF(JPSL.EQ.0) GO TO 300
      JIP=IPSL(LS1)
      JLP=JIP+7
```

```
          DO 230 JPS=1,JPSL
          JLIN=LS1
          JSQT=0
              DO 220 JLT=1,8
                  DO 210 JPIX=JIP,JLP
                  JSQT=JSQT+1
210               ISF(JSQT)=IBUF(JPIX,JLIN)
              JLIN=JLIN+1
              IF(JLIN.GT.8) JLIN=1
220           CONTINUE
          JREC=JREC+1                                     !  I*4
              WRITE (3'JREC) (ISF(JSQT),JSQT=1,KSQT)
          JIP=JIP+64
230       JLP=JLP+64
300   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE LINEIN
C
C  Purpose: Input next line from source image raster input file
C               Pad line to integral number of subframes
C               Convert pixels from 6 bits to 8 bits if designated
C               Skip designated number of input pixels
C
C  CALLed from: SQUARE,STAGGR
C
C-----------------------------------------------------------------------
      COMMON IBUF(4032,8)
      BYTE IBUF
      COMMON /LINEUP/ IP,IPR,LP,NBT,L8,LK,LSKPT
      COMMON /SFTEMP/ NLT,NPT,KSFT,KSQT,NST
      INTEGER*4 NST
      DIMENSION KPX6(64)
      DATA KPX6/0,"4,"10,"14,"20,"24,"30,"34,"40,"44,"50,"54,"60,"64,
     +  "70,"74,"100,"104,"110,"114,"120,"124,"130,"134,"140,"144,
     +  "150,"154,"160,"164,"170,"174,"200,"204,"210,"214,"220,"224,
     +  "230,"234,"240,"244,"250,"254,"260,"264,"270,"274,"300,"304,
     +  "310,"314,"320,"324,"330,"334,"340,"344,"350,"354,"360,"364,
     +  "370,"374/
      READ (2,END=110,ERR=110) (IBUF(J,L8),J=1,IPR)
      GO TO 120
110         TYPE 500,LK,LSKPT
500         FORMAT(/,1X,'*** EOF/ERR AT LINE'I5,' (INCLUDING',I5,
     +        ' SKIPS)')
            STOP
120   DO 130 J=IPR,LP
130   IBUF(J+1,L8)=IBUF(J,L8)
      IPIX=IP
         DO 140 JPIX=1,NPT
         KPIX=IBUF(IPIX,L8)
         IPIX=IPIX+1
         IF(NBT.EQ.0) GO TO 140
         KP6=KPIX.AND."77
         KPIX=KPX6(KP6+1)
140      IBUF(JPIX,L8)=KPIX
150   RETURN
      END
```

SECTION EIGHT

TRANSMAPS MODULE #2: MAPS COMPRESSION

8.1 Program Characteristics

Program Names: MAPS or MP

Subroutines:	FILESM USERM SETKCM SETUPM RMPSET ZIGZAG LVLSET SFMAC SFIN SFMAPS QDIFF THRESH MAPOUT LSTREC SUMMRY	
Files:	MSET.DAT IMAGE.DAT MAPS.DAT	(input/output) (input) (output)

Task Build Options: MAXBUF = 1024

Task Size: 31488

8.2 Source Listing

The COMMENT-annotated source listing for MAPS follows:

```
      PROGRAM MAPS
C
C  +-----------------------------------------+
C  !                                         !
C  !  TransMAPS Module #2:  MAPS Compression  !
C  !                                         !
C  +-----------------------------------------+
C
C                                        Control Data Corporation - 1982
C
C Files: Unit  Name    Content                          From/To  Type
C ------
C In/Out  1    MSET    User-defined parameter set       MAPS     SEGMENTED
C   In    2    IMAGE   Source image, subframes          SUBFRM   FIXED SEQ.
C   Out   3    MAPS    MAPSel stream, 512 byte rec.     DMAPS    DIRECT
C
C User Interaction:  In Subroutine USERM
C -----------------
C      Macro-fidelity Control
C      Micro-fidelity Control
C      Gray-scale Manipulations
C
C Program Structure:
C ------------------
C      PROGRAM MAPS
C          CALL FILESM   Open files, read file header for source image
C          CALL USERM    Specify macro-fidelity, micro-fidelity, and
C                                                  gray-scale controls
C              CALL SETKCM(K)   Establish contrast control matrix, Set K
C          CALL SETUPM   Establish compression control tables
C              CALL RMPSET   Establish contrast and intensity mappings
C              CALL ZIGZAG   Establish raster to zigzag reorder table
C              CALL LVLSET   Establish 3 MAPSel/byte level packing
C          Loop on subframes (LOOPSF)
C              CALL SFMAC(LOOPSF)   Determine macro-fidelity position
C              CALL SFIN   Input subframe, map gray scales, zigzag order
C              CALL SFMAPS   MAPS subframe compression kernel
C                  Loop on levels (resolution)
C                      Loop on MAPSel quads
C                          CALL QDIFF   Form contrasts, sign-sort vector
C                          CALL THRESH   Test contrasts, form MAPSel
C                  Transfer completed MAPSels to output buffer
C                      CALL MAPOUT   Output MAPSel stream record
C          CALL LSTREC   Output final (partial) MAPSel stream record
C          CALL SUMMRY   Determine optimal biases, compression, & MSE
C
C COMMON Block Communication:
C ---------------------------
C    /BLKPAT/   Block/pattern mode by level           Length: 21
C                   SETUPM,SFMAPS
C
C      LBPT(5)         [I*2]   Block(0)/pattern(1) for levels 1-5
C      IQBP(4,4)       [I*2]   Low/High index by Quadrant,Pattern
C
```

```
C     /CNTRST/   Functional contrast control matrix      Length: 81
C                    SETUPM,SFMAC,SFMAPS,THRESH
C
C        KCMT(80)         [I*2]    Sequentially-addressed contrast matrix
C        KINDEX           [I*2]    Pointer for current macro-fidelity index
C
C     /CONTRL/   User-interactive input specifications   Length: 415
C                    USERM,SETKCM,SETUPM,RMPSET,SFMAC
C
C        MAC(16,16)       [I*2]    Macro-fidelity image partition
C        KCS(4)           [I*2]    Micro-type: parametric(0)/matrix(1)
C        CS(4)            [R*4]    Contrast scale parameter
C        TB(4)            [R*4]    Taper base parameter
C        SF(4)            [R*4]    Step fraction parameter
C        SB(4)            [R*4]    Step bias parameter
C        KCM(4,5,4)       [I*2]    Contrast control matrices
C                                    (Contrast,Transition,Macro-group)
C        LBP(6)           [I*2]    Block(0)/pattern(1) mode (Level+1)
C        KBP(2,9)         [I*2]    Contrast-space remap breakpoint pairs
C        IBP(2,9)         [I*2]    Intensity-space remap breakpoint pairs
C        IRSET            [I*2]    Intensity reset type (N P L S T H)
C
C     /GRYSCL/   Gray-scale remap tables                 Length: 4608
C                    SETUPM,RMPSET,SFIN,SFMAPS,THRESH
C
C        KRMP(256)        [I*2]    Code to Contrast space remap
C        IRMP(256)        [I*2]    Code to Intensity space remap
C        IDMP(4096)       [I*2]    Intensity to Code space demap
C
C     /HEADER/   Standard MAPS file header               Length: 32
C                    MAPS,FILESM,USERM,SETUPM,SUMMRY
C
C        IFILE            [I*2]    File type
C        INAME(8)         [Byte]   User-selected image name
C        NL               [I*2]    Number of lines in source image
C        NP               [I*2]    Number of pixels in source image
C        NB               [I*2]    Number of bits/pixel in source image
C        KSF              [I*2]    Kind of subframe 8x8 16x16 32x32
C        IGRD             [I*2]    Subframe grid: square(0)/staggered(1)
C        NS               [I*4]    Total subframe count
C        MC               [I*4]    MAPSel count
C        MIXBP            [I*2]    Packed block(0)/pattern(1) mode (rt-lft)
C        IBV(5,2)         [I*2]    Optimal pattern biases by level,low/high
C        IPAD(7)          [I*2]    Space for future extension
C
C     /LVLTBL/   Resolution (level) code packing table   Length: 366
C                    SETUPM,LVLSET,MAPOUT,LSTREC
C
C        LVLT(366)        [I*2]    Level code triplet to byte conversion
C
```

```
C     /MAPSSF/    MAPS subframe data spaces                   Length: 4098
C                      SETUPM,SFIN,SFMAPS,THRESH
C
C        NSQ                 [I*2]    Number of total pixels/subframe
C        NLVL                [I*2]    Number of active levels (4 5 6)
C        ICODE(1024)         [I*2]    Subframe pixels in code (8 bit) space
C        KNTRST(1024)        [I*2]    Pixel/MAPSel remap to contrast space
C        INTENS(1024)        [I*2]    Pixel/MAPSel remap to intensity space
C        LEVEL(1024)         [Byte]   MAPSel resolution (level) code
C        PATTRN(1024)        [Byte]   MAPSel pattern code
C
C     /MDATA/     MAPS output buffer                          Length: 259
C                      MAPS,FILESM,SETUPM,SFMAPS,
C                      MAPOUT,LSTREC,SUMMRY
C
C        MREC                [I*4]    Index of most recently written record
C        MLOC                [I*2]    Index of most recent buffer entry
C        MSF(512)            [Byte]   MAPSel stream assembly buffer
C
C     /MSTATS/    Accumulators for optimum bias & MSE         Length: 80
C                      SETUPM,SFMAPS,SUMMRY
C
C        KOUNT(2,6)          [I*4]    Pixel count by low/high,level
C        DIFF(2,6)           [R*8]    Sum of (I-M) by low/high,level
C        DIFFSQ              [R*8]    Sum square of (I-M) [source-MAPS]
C        SUMSQ               [R*8]    Sum square of I [source intensities]
C
C     /MTEMP/     Temporary staging, partial multiplets       Length: 11
C                      SETUPM,SFMAPS,MAPOUT,LSTREC
C
C        NM                  [I*2]    Multiplet size (4 for 8x8, 3 for 16, 32)
C        KM                  [I*2]    Count currently in list
C        MAPSEL(5)           [I*2]    MAPS intensity/pattern values
C        MLVL(4)             [I*2]    MAPS resolution (level) codes
C
C     /QUAD/      Current quad of MAPS elements               Length: 17
C                      SFMAPS,QDIFF,THRESH
C
C        KT(4)               [I*2]    Contrast space quad
C        IT(4)               [I*2]    Intensity space quad
C        NT(6)               [I*2]    Contrasts: 3-2, 3-1, 2-0, 1-0, 2-1, 3-0
C        LQ                  [I*2]    Location of quad (zigzag index of start)
C        LVLP                [I*2]    Level resulting if quad is combined
C        NN                  [I*2]    Contrast-sign sort vector
C
C     /RESET/     Intensity reset controls                    Length: 6
C                      SETUPM,RMPSET,THRESH
C
C        MSK(4)              [I*2]    Activation masks by sort order
C        NORM                [I*2]    Normalization divisor (4 2 1)
C        NBIAS               [I*2]    Bias for rounding (2 1 0)
C
```

```
C    /SFCNTL/   Subframe index to Macro-fidelity index  Length: 25
C                   SETUPM,ZIGZAG,SFMAC
C
C       KSFT            [I*2]   Subframe size: Edge (8 16 32)
C       KSQT            [I*2]   Subframe size: Pixel count
C       NPST            [I*2]   Number of subframes in pixel direction
C       NPSL(8)         [I*2]   Number of staggered subframes in pixel
C                                 direction for each startline mod 8
C       MPSL(8)         [I*2]   Middle pixel of first staggered
C                                 subframe for each startline mod 8
C       KSFH            [I*2]   Subframe half size
C       FL              [R*4]   Line to Macro-fidelity index factor
C       FP              [R*4]   Pixel to Macro-fidelity index factor
C       JGRD            [I*2]   Grid type: square(0)/staggered(1)
C
C    /SFDATA/   Source image subframe data             Length: 512
C                   SFIN
C
C       ISF(1024)       [Byte]  Subframe input array
C
C    /ZIGZAG/   Subframe raster to zigzag conversion    Length: 1024
C                   SETUPM,ZIGZAG,SFIN
C
C       IZZ(1024)       [I*2]   Raster to zigzag lookup table
C
C Zigzag Order Convention:
C -----------------------
C                                  Pixel direction
C                                      --->
C                                  +-----------+
C                                  |     |     |
C                                  |  0  |  1  |
C                                  |     |     |
C          Line direction      |   |-----+-----|
C                              |   |     |     |
C                              V   |  2  |  3  |  MAPSel Quadrants
C                                  |     |     |
C                                  +-----------+
C
C---------------------------------------------------------------------
      COMMON /HEADER/ IFILE,INAME(8),NL,NP,NB,KSF,IGRD,NS,MC,MIXBP,
     +   IBV(5,2),IPAD(7)
      BYTE INAME
      INTEGER*4 NS,MC
          DIMENSION JHEAD(32)
          EQUIVALENCE (JHEAD(1),IFILE)
      COMMON /MDATA/ MREC,MLOC,MSF(512)
      BYTE MSF
      INTEGER*4 MREC
          DIMENSION JSF(256)
          EQUIVALENCE (JSF(1),MSF(1))
      INTEGER*4 LOOPSF
      CALL FILESM
      CALL USERM
```

```
      TYPE 500,INAME,NL,NP
500   FORMAT(/,1X,'MAPS COMPRESSING IMAGE ',8A1,',',I5,' LINES BY',
     +  I5,' PIXELS')
      CALL SETUPM
      DO 110 LOOPSF=1,NS                                   !  I*4
      CALL SFMAC(LOOPSF)
      CALL SFIN
      CALL SFMAPS
110   CONTINUE
      CALL LSTREC
      CALL SUMMRY
      DO 120 J=1,256
120   JSF(J)=0
      DO 130 J=1,32
130   JSF(J)=JHEAD(J)
      WRITE (3'1) JSF
      CLOSE(UNIT=2)
      CLOSE(UNIT=3)
      END
```

```
      SUBROUTINE FILESM
C
C  Purpose: Open files, read input header, write preliminary output
C           header
C
C  CALLed from: MAPS
C
C-----------------------------------------------------------------------
      COMMON /HEADER/ IFILE,INAME(8),NL,NP,NB,KSF,IGRD,NS,MC,MIXBP,
     +  IBV(5,2),IPAD(7)
      BYTE INAME
      INTEGER*4 NS,MC
          DIMENSION JHEAD(32)
          EQUIVALENCE (JHEAD(1),IFILE)
      COMMON /MDATA/ MREC,MLOC,MSF(512)
      BYTE MSF
      INTEGER*4 MREC
          DIMENSION JSF(256)
          EQUIVALENCE (JSF(1),MSF(1))
      OPEN(UNIT=2,TYPE='OLD',NAME='IMAGE',FORM='UNFORMATTED',
     +  RECORDTYPE='FIXED')
      READ (2) JHEAD
      OPEN(UNIT=3,TYPE='NEW',NAME='MAPS',FORM='UNFORMATTED',
     +  RECORDTYPE='FIXED',RECORDSIZE=128,ACCESS='DIRECT')
      DO 110 J=1,256
110   JSF(J)=0
      DO 120 J=1,32
120   JSF(J)=JHEAD(J)
      WRITE (3'1) JSF
      RETURN
      END
```

```
      SUBROUTINE USERM
C
C  Purpose: User interaction for MAPS compression
C           Mode:  Quick User Full
C           MAPS macro-fidelity control
C           MAPS micro-fidelity control
C           MAPS gray-scale manipulations
C
C  CALLed from: MAPS
C
C  CALLs: SETKCM
C
C-------------------------------------------------------------------
      COMMON /CONTRL/ MAC(16,16),KCS(4),CS(4),TB(4),SF(4),SB(4),
     +  KCM(4,5,4),LBP(6),KBP(2,9),IBP(2,9),IRSET
      COMMON /HEADER/ IFILE,INAME(8),NL,NP,NB,KSF,IGRD,NS,MC,MIXBP,
     +  IBV(5,2),IPAD(7)
      DIMENSION MSET(415)
      EQUIVALENCE (MSET,MAC)
      BYTE INAME
      INTEGER*4 NS,MC
      DIMENSION MT(16),IA(4)
      DATA CSD,TBD,SFD,SBD/20.0,3.0,0.5,0.1/                ! Default
      DATA LBP2,LBP3,LBP4,LBP5,LBP6/1,1,0,0,0/              ! Default
      DATA MAXK,MAXI/4095,4095/
      LVL=4
      IF(KSF.EQ.16) LVL=5
      IF(KSF.EQ.32) LVL=6
      LT=LVL-1
      DO 10 L=1,16
      DO 10 I=1,16
10    MAC(I,L)=1                                            ! Default
      DO 20 K=1,4
      KCS(K)=1HP
      CS(K)=CSD
      TB(K)=TBD
      SF(K)=SFD
      SB(K)=SBD
      CALL SETKCM(K)
20    CONTINUE
      LBP(1)=0
      LBP(2)=LBP2
      LBP(3)=LBP3
      LBP(4)=LBP4
      LBP(5)=LBP5
      LBP(6)=LBP6
      KBP(1,1)=0
      KBP(2,1)=0
      IBP(1,1)=0
      IBP(2,1)=0
      DO 30 J=2,9
      KBP(1,J)=255                                          ! Default
      KBP(2,J)=255                                          ! Default
      IBP(1,J)=255                                          ! Default
30    IBP(2,J)=255                                          ! Default
      IRSET=1HM                                             ! Default
```

```
      TYPE 500
500   FORMAT(/,1X,27(1H*),/,1X,'*',25X,'*',/;1X,
     +  '* MAPS COMPRESSION MODULE *',
     +  /,1X,'*',25X,'*',/,1X,27(1H*))
      TYPE 510
510   FORMAT(/,3X,'USER OPTION MODES:',
     +  //,5X,'Q - QUICK MODE (SELECT CONTRAST SCALE ONLY)',
     +  /,5X,'U - USER PRE-DEFINED PARAMETERS FROM FILE MSET.DAT',
     +  /,5X,'F - FULL OPTION REVIEW AND SELECTIVE REVISION',
     +  //,5X,'MODE? (Q  U  F)  Q')
      ACCEPT 1,LIT
1     FORMAT(10A1)
      IF(LIT.EQ.1HF) GO TO 100
      IF(LIT.EQ.1HU) GO TO 80
      TYPE 520,CS(1)
520   FORMAT(/,5X,'CONTRAST SCALE?',F7.1,5X,'(/ = NO CHNG)')
      ACCEPT *,CS(1)
      CALL SETKCM(1)
      GO TO 400
80    OPEN(UNIT=1,TYPE='OLD',NAME='MSET',FORM='UNFORMATTED',ERR=90)
      READ (1) MSET
      CLOSE(UNIT=1)
      GO TO 400
90    TYPE 530
530   FORMAT(/,1X,'*** NO PRE-DEFINED PARAMETER FILE FOUND: ',
     +  'SET DIRECTLY')
C
C *** MAPS Macro-fidelity Control
C
100   TYPE 540
540   FORMAT(//,3X,'MACRO-FIDELITY CONTROL:  REVIEW/REVISE? ',
     +  '(Y OR N)  N')
      ACCEPT 1,LIT
      IF(LIT.NE.1HY) GO TO 140
110   TYPE 550,((MAC(I,L),I=1,16),L,L=1,16)
550   FORMAT(/,12X,'CURRENT IMAGE PARTITION',/,
     +  16(/,8X,16I2,5X,'ROW',I3),//,5X,'ROW TO CHANGE? (1-16)',
     +  5X,'(/ = NO FURTHER CHNG)')
      L=0
      ACCEPT *,L
      IF((L.LT.1).OR.(L.GT.16)) GO TO 140
      DO 120 I=1,16
120   MT(I)=MAC(I,L)
      TYPE 560,L,MT
560   FORMAT(3X,'REVISE ROW',I2,'?  (RANGE: 1-4)     (/ = NO CHNG)',
     +  /,16(1X,I1))
      ACCEPT *,MT
      DO 130 I=1,16
      IT=MT(I)
      IF((IT.GE.1).AND.(IT.LE.4)) MAC(I,L)=IT
130   CONTINUE
      GO TO 110
140   DO 150 K=1,4
150   IA(K)=0
      DO 160 L=1,16
      DO 160 I=1,16
      IT=MAC(I,L)
160   IA(IT)=IA(IT)+1
```

```
C
C ***   MAPS Micro-fidelity Control
C
        TYPE 570
570     FORMAT(/,3X,'MICRO-FIDELITY CONTROL:  REVIEW/REVISE? ',
     +    '(Y OR N)  N')
        ACCEPT 1,LIT
        IF(LIT.NE.1HY) GO TO 300
200     DO 270 K=1,4
        IF(IA(K).LE.0) GO TO 270
210     IF(KCS(K).EQ.1HM) GO TO 220
        TYPE 580,K,CS(K),TB(K),SF(K),SB(K)
580     FORMAT(7X,'GROUP',I2,' CONTRAST THRESHOLD PARAMETERS',
     +    /,9X,'CONTRAST SCALE',F7.1,/,9X,'TAPER',9X,F7.1,
     +    /,9X,'STEP FRACTION ',F7.1,/,9X,'STEP BIAS     ',F7.1)
220     TYPE 590,K,(L-1,L,(KCM(J,L,K),J=1,4),L,L=1,LT)
590     FORMAT(/,7X,'GROUP',I2,' CONTRAST THRESHOLD MATRIX',
     +    /,16X,'E    M    L    U',5(/,9X,I1,'-',I1,4I5,5X,'ROW',I2))
230     TYPE 600
600     FORMAT(/,5X,'SPECIFICATION MODES:',/,7X,'N - NO CHANGE',
     +    /,7X,'S - SCALE ONLY',/,7X,'P - PARAMETRIC',/,7X,'M - MATRIX',
     +    //,5X,'REVISE SPECIFICATIONS? (N S P M)  N')
        ACCEPT 1,LIT
        IF(LIT.NE.1HM) GO TO 250
        KCS(K)=1HM
240     L=0
        TYPE 610,LT
610     FORMAT(/,5X,'MATRIX ROW TO CHANGE?  (1-',I1,')',5X,
     +    '(/ = NO FURTHER CHNG)')
        ACCEPT *,L
        IF((L.LT.1).OR.(L.GT.LT)) GO TO 270
        TYPE 620,K,L-1,L,(KCM(J,L,K),J=1,4)
620     FORMAT(2X,'REVISE GROUP',I2,'/LEVEL ',I1,'-',I1,'?',
     +    /,4X,'E    M    L    U',/,4I5,5X,'(/ = NO CHNG)')
        ACCEPT *,(KCM(J,L,K),J=1,4)
        TYPE 590,K,(L-1,L,(KCM(J,L,K),J=1,4),L,L=1,LT)
        GO TO 240
250     IF((LIT.NE.1HS).AND.(LIT.NE.1HP)) GO TO 270
        KCS(K)=1HP
        TYPE 630,K,CS(K)
630     FORMAT(5X,'GROUP',I2,' CONTRAST SCALE?',F7.1,5X,
     +    '(/ = NO CHNG)')
        ACCEPT *,CS(K)
        IF(LIT.EQ.1HS) GO TO 260
        TYPE 640,K,TB(K)
640     FORMAT(5X,'GROUP',I2,' TAPER?',F5.1,5X,'(/ = NO CHNG)')
        ACCEPT *,TB(K)
        TYPE 650,K,SF(K)
650     FORMAT(5X,'GROUP',I2,' STEP FRACTION?',F4.1,5X,'(/ = NO CHNG)')
        ACCEPT *,SF(K)
        TYPE 660,K,SB(K)
660     FORMAT(5X,'GROUP',I2,' STEP BIAS?',F4.1,5X,'(/ = NO CHNG)')
        ACCEPT *,SB(K)
260     CALL SETKCM(K)
        GO TO 210
270     CONTINUE
```

```
      DO 280 L=1,LVL
      MT(L)=1HB
      IF(LBP(L).NE.0) MT(L)=1HP
280   CONTINUE
      TYPE 670,(L-1,L=1,LVL)
670   FORMAT(/,3X,'BLOCK/PATTERN ASSIGNMENT:',//,9X,'LEVEL  ',6I1)
      TYPE 680,(MT(L),L=1,LVL)
680   FORMAT(9X,'MODE   ',6A1)
      TYPE 690,(MT(L),L=1,LVL)
690   FORMAT(/,5X,'REVISE B/P VECTOR? ',6A1)
      ACCEPT 1,(MT(L),L=1,LVL)
      DO 290 L=2,LVL
      IF(MT(L).EQ.1HB) LBP(L)=0
      IF(MT(L).EQ.1HP) LBP(L)=1
290   CONTINUE
C
C ***   MAPS Gray-scale Manipulations
C
300   TYPE 700
700   FORMAT(/,3X,'GRAY-SCALE MANIPULATIONS:  REVIEW/REVISE? ',
     +  '(Y OR N)  N')
      ACCEPT 1,LIT
      IF(LIT.NE.1HY) GO TO 400
310   DO 320 K=2,9
      IF(KBP(1,K).GE.255) GO TO 330
320   CONTINUE
      K=9
330   TYPE 710,(KBP(1,J),KBP(2,J),J=1,K)
710   FORMAT(/,5X,'CONTRAST SPACE REMAPPING: PIECEWISE LINEAR',//,
     +  7X,'(CODE SPACE/CONTRAST SPACE) BREAKPOINT PAIRS',//,
     +  (11X,I3,6X,I4))
      TYPE 720
720   FORMAT(/,5X,'REVISE CONTRAST REMAP? (Y OR N)  N')
      ACCEPT 1,LIT
      IF(LIT.NE.1HY) GO TO 350
      DO 340 K=2,9
      K1L=KBP(1,K-1)
      IF(K.EQ.9) K1L=255
      K2L=KBP(2,K-1)
      K1U=255
      K2U=MAXK
      K1T=KBP(1,K)
      IF(K1T.LE.K1L) K1T=K1L
      K2T=KBP(2,K)
      IF(K2T.LE.K2L) K2T=K2L
      TYPE 730,K,K1L,K1U,K2L,K2U,K,K1T,K2T
730   FORMAT(/,7X,'POINT',I2,':',/,9X,'CODE SPACE RANGE',6X,
     +  '(',I3,'-',I3,')',/,9X,'CONTRAST SPACE RANGE (',I4,'-',I4,')',
     +  //,5X,'REVISE',I2,' (CODE/CONTRAST)?',I4,I5,'  (/ = NO CHNG)')
      ACCEPT *,K1T,K2T
      IF(K1T.LE.K1L) K1T=K1L
      IF(K1T.GT.K1U) K1T=K1U
      KBP(1,K)=K1T
      IF(K2T.LE.K2L) K2T=K2L
      IF(K2T.GT.K2U) K2T=K2U
      KBP(2,K)=K2T
      IF(K1T.EQ.255) GO TO 310
340   CONTINUE
```

```
      GO TO 310
350   DO 360 I=2,9
      IF(IBP(1,I).GE.255) GO TO 370
360   CONTINUE
      I=9
370   TYPE 740,(IBP(1,J),IBP(2,J),J=1,I)
740   FORMAT(/,5X,'INTENSITY SPACE REMAPPING: PIECEWISE LINEAR',//,
     +  7X,'(CODE SPACE/INTENSITY SPACE) BREAKPOINT PAIRS',//,
     +  (11X,I3,6X,I4))
      TYPE 750
750   FORMAT(/,5X,'REVISE INTENSITY REMAP? (Y OR N)  N')
      ACCEPT 1,LIT
      IF(LIT.NE.1HY) GO TO 390
      DO 380 I=2,9
      I1L=IBP(1,I-1)
      IF(I.EQ.9) I1L=255
      I2L=IBP(2,I-1)
      I1U=255
      I2U=MAXI
      I1T=IBP(1,I)
      IF(I1T.LE.I1L) I1T=I1L
      I2T=IBP(2,I)
      IF(I2T.LE.I2L) I2T=I2L
      TYPE 760,I,I1L,I1U,I2L,I2U,I,I1T,I2T
760   FORMAT(/,7X,'POINT',I2,':',/,9X,'CODE SPACE RANGE',7X,
     +  '(',I3,'-',I3,')',/,9X,'INTENSITY SPACE RANGE (',I4,'-',I4,
     +  ')',//,5X,'REVISE',I2,' (CODE/INTENSITY)?',I4,I5,
     +  '  (/ = NO CHNG)')
      ACCEPT *,I1T,I2T
      IF(I1T.LE.I1L) I1T=I1L
      IF(I1T.GT.I1U) I1T=I1U
      IBP(1,I)=I1T
      IF(I2T.LE.I2L) I2T=I2L
      IF(I2T.GT.I2U) I2T=I2U
      IBP(2,I)=I2T
      IF(I1T.EQ.255) GO TO 350
380   CONTINUE
      GO TO 350
390   TYPE 770,IRSET
770   FORMAT(/,5X,'INTENSITY RESET:',//,7X,'M - MEAN OF QUAD',/,7X,
     +  'P - PSEUDO-MEDIAN OF QUAD',/,7X,'L - LOWEST IN QUAD',/,7X,
     +  'S - SECOND IN QUAD',/,7X,'T - THIRD IN QUAD',/,7X,
     +  'H - HIGHEST IN QUAD',//,5X,'REVISE RESET? (M P L S T H) ',
     +  A1)
      ACCEPT 1,LIT
      IF(LIT.EQ.1HM) IRSET=1HM
      IF(LIT.EQ.1HP) IRSET=1HP
      IF(LIT.EQ.1HL) IRSET=1HL
      IF(LIT.EQ.1HS) IRSET=1HS
      IF(LIT.EQ.1HT) IRSET=1HT
      IF(LIT.EQ.1HH) IRSET=1HH
```

```
400     TYPE 780
780     FORMAT(//,3X,'USER SPECIFICATION COMPLETE:',/,3X,28(1H*),//,5X,
     +    'REVIEW? (Y OR N)  N')
        ACCEPT 1,LIT
        IF(LIT.EQ.1HY) GO TO 100
        TYPE 790
790     FORMAT(3X,'SAVE THESE PARAMETERS FOR FUTURE USE? (Y OR N) N')
        ACCEPT 1,LIT
        IF(LIT.NE.1HY) GO TO 410
        OPEN(UNIT=1,TYPE='NEW',NAME='MSET',FORM='UNFORMATTED')
        WRITE (1) MSET
        ENDFILE 1
        CLOSE(UNIT=1)
        TYPE 800
800     FORMAT(5X,'PARAMETERS SAVED ON FILE MSET.DAT')
410     CONTINUE
        RETURN
        END
```

```
      SUBROUTINE SETKCM(K)
C
C  Purpose: Infer Group K contrast control matrix from parametric
C           specification
C
C  CALLed from: USERM
C
C-----------------------------------------------------------------
      COMMON /CONTRL/ MAC(16,16),KCS(4),CS(4),TB(4),SF(4),SB(4),
     +   KCM(4,5,4),LBP(6),KBP(2,9),IBP(2,9),IRSET
      TE=CS(K)
      B=TB(K)
      FM=SF(K)
      FO=FM+SB(K)
      IF(B.GT.0.) GO TO 10
      TYPE 100,B,K
100   FORMAT(/,'***** GROUP',I2,' TAPER =',F8.1)
10    DO 20 L=1,5
      IT=TE+0.5
      KCM(1,L,K)=IT
      IT=FM*TE+0.5
      KCM(2,L,K)=IT
      IT=FO*TE+0.5
      KCM(3,L,K)=IT
      KCM(4,L,K)=IT
20    TE=TE/B
      RETURN
      END
```

```
      SUBROUTINE SETUPM
C
C  Purpose: Establish MAPS compression controls
C                Pack block(0)/pattern(1) modes to bit vector by level
C                Establish subframe partition parameters for macro-fidelity
C                  index determination
C                Establish contrast control matrices with single index
C                  addressing
C                Establish contrast remap and intensity remap/demap tables
C                Establish subframe raster to zigzag conversion
C                Establish level-triplet to byte resolution packing table
C                Initialize optimum bias and performance evaluation
C                  accumulators
C
C  CALLed from: MAPS
C
C  CALLs: RMPSET,ZIGZAG,LVLSET
C
C-----------------------------------------------------------------------
      COMMON /CONTRL/ MAC(16,16),KCS(4),CS(4),TB(4),SF(4),SB(4),
     +  KCM(4,5,4),LBP(6),KBP(2,9),IBP(2,9),IRSET
      COMMON /HEADER/ IFILE,INAME(8),NL,NP,NB,KSF,IGRD,NS,MC,MIXBP,
     +  IBV(5,2),IPAD(7)
      BYTE INAME
      INTEGER*4 NS,MC
      COMMON /SFCNTL/ KSFT,KSQT,NPST,NPSL(8),MPSL(8),KSFH,FL,FP,JGRD
            DIMENSION JPSL(8)
      COMMON /MDATA/ MREC,MLOC,MSF(512)
      BYTE MSF
      INTEGER*4 MREC
      COMMON /MTEMP/ NM,KM,MAPSEL(5),MLVL(4)
      COMMON /CNTRST/ KCMT(80),KINDX
      COMMON /BLKPAT/ LBPT(5),IQBP(4,4)
      COMMON /GRYSCL/ KRMP(256),IRMP(256),IDMP(4096)
      COMMON /RESET/ MSK(4),NORM,NBIAS
      COMMON /ZIGZAG/ IZZ(1024)
      COMMON /LVLTBL/ LVLT(366)
      COMMON /MAPSSF/ NSQ,NLVL,ICODE(1024),KNTRST(1024),INTENS(1024),
     +  LEVEL(1024),PATTRN(1024)
      BYTE LEVEL,PATTRN
      COMMON /MSTATS/ KOUNT(2,6),DIFF(2,6),DIFFSQ,SUMSQ
      INTEGER*4 KOUNT
      REAL*8 DIFF,DIFFSQ,SUMSQ
      DATA NPSL/8*0/,MPSL/28,52,12,36,60,20,44,4/
            DATA JPSL/4,1,6,3,0,5,2,7/
      DATA IQBP/1,1,2,2,2,1,2,1,1,2,1,2,2,2,1,1/
      IFILE=1
      MIXBP=0
      M=1
      DO 110 J=1,6
      MIXBP=MIXBP.OR.(M*(LBP(J).AND."1))
110   M=2*M
      KSFT=KSF
      KSQT=KSF*KSF
      NPST=(NP-1)/KSF+1
```

```
      IF(IGRD.EQ.0) GO TO 130
      DO 120 J=1,8
120   NPSL(J)=(NPST+JPSL(J))/8
130   KSFH=KSF/2
      FL=16./NL
      FP=16./NP
      JGRD=IGRD
      MREC=1
      MLOC=0
      DO 140 J=1,512
140   MSF(J)=0
      NM=4
      IF(KSF.NE.8) NM=3
      KM=0
      I=0
      DO 170 L=1,4
          DO 160 K=1,5
              DO 150 J=1,4
              I=I+1
150           KCMT(I)=KCM(J,K,L)
160       CONTINUE
170   CONTINUE
      KINDX=1
      DO 180 J=1,5
180   LBPT(J)=LBP(J+1)
      CALL RMPSET
      CALL ZIGZAG
      IF(KSF.GT.8) CALL LVLSET
      NSQ=KSF*KSF
      NLVL=4
      IF(KSF.EQ.16) NLVL=5
      IF(KSF.EQ.32) NLVL=6
      DO 190 J=1,1024
      ICODE(J)=0
      KNTRST(J)=0
      INTENS(J)=0
      LEVEL(J)=0
      PATTRN(J)=0
190   CONTINUE
      DO 210 J=1,6
          DO 200 I=1,2
          KOUNT(I,J)=0
200       DIFF(I,J)=0.D0
210   CONTINUE
      DIFFSQ=0.D0
      SUMSQ=0.D0
      RETURN
      END
```

```
      SUBROUTINE RMPSET
C
C  Purpose: Establish functional gray-scale manipulation controls
C               Establish code (8-bit gray-scale) to contrast space remap
C               Establish code to intensity space remap and demap
C               Establish intensity reset masks, normalization factor,
C                 and rounding bias
C
C  CALLed from: SETUPM
C
C-----------------------------------------------------------------------
      COMMON /CONTRL/ MAC(16,16),KCS(4),CS(4),TB(4),SF(4),SB(4),
     +  KCM(4,5,4),LBP(6),KBP(2,9),IBP(2,9),IRSET
      COMMON /GRYSCL/ KRMP(256),IRMP(256),IDMP(4096)
      COMMON /RESET/ MSK(4),NORM,NBIAS
      DO 130 JSEG=2,9
      LBP1=KBP(1,JSEG-1)+1
      LBP2=KBP(1,JSEG)+1
      DL=LBP2-LBP1
      NBP1=KBP(2,JSEG-1)+1
      NBP2=KBP(2,JSEG)+1
      DN=NBP2-NBP1
      IF(LBP1.EQ.LBP2) GO TO 120
      F=DN/DL
          DO 110 J=LBP1,LBP2
110       KRMP(J)=NBP1+F*(J-LBP1)-0.5
120   KRMP(LBP1)=NBP1-1
      IF(LBP2.GE.256) GO TO 200
130   CONTINUE
200   DO 250 JSEG=2,9
      LBP1=IBP(1,JSEG-1)+1
      LBP2=IBP(1,JSEG)+1
      DL=LBP2-LBP1
      NBP1=IBP(2,JSEG-1)+1
      NBP2=IBP(2,JSEG)+1
      DN=NBP2-NBP1
      IF(LBP1.EQ.LBP2) GO TO 220
      F=DN/DL
          DO 210 J=LBP1,LBP2
210       IRMP(J)=NBP1+F*(J-LBP1)-0.5
220   IRMP(LBP1)=NBP1-1
      IF(NBP1.EQ.NBP2) GO TO 240
      F=DL/DN
          DO 230 J=NBP1,NBP2
230       IDMP(J)=LBP1+F*(J-NBP1)-0.5
240   IDMP(NBP1)=LBP1-1
      IF(LBP2.GE.256) GO TO 300
250   CONTINUE
```

```
300     DO 310 J=1,4
310     MSK(J)=0
        IR=IRSET
        IF((IR.EQ.1HM).OR.(IR.EQ.1HL)) MSK(1)="7777
        IF((IR.EQ.1HM).OR.(IR.EQ.1HP).OR.(IR.EQ.1HS)) MSK(2)="7777
        IF((IR.EQ.1HM).OR.(IR.EQ.1HP).OR.(IR.EQ.1HT)) MSK(3)="7777
        IF((IR.EQ.1HM).OR.(IR.EQ.1HH)) MSK(4)="7777
        NORM=1
        IF(IR.EQ.1HP) NORM=2
        IF(IR.EQ.1HM) NORM=4
        NBIAS=NORM/2
        RETURN
        END
```

```
      SUBROUTINE ZIGZAG
C
C  Purpose: Establish subframe raster to zigzag conversion table
C
C  CALLed from: SETUPM
C
C-------------------------------------------------------------------
      COMMON /SFCNTL/ KSFT,KSQT,NPST,NPSL(8),MPSL(8),KSFH,FL,FP,JGRD
      COMMON /ZIGZAG/ IZZ(1024)
      DATA IZZ/1024*0/
      DO 120 JZZ=1,KSQT
      MZZ=JZZ-1
      KMSK=MZZ.AND."525
      LMSK=(MZZ/2).AND."525
      K=0
      L=0
      M=1
          DO 110 J=1,5
          KT=KMSK.AND."1
          K=K.OR.(M*KT)
          LT=LMSK.AND."1
          L=L.OR.(M*LT)
          KMSK=KMSK/4
          LMSK=LMSK/4
110       M=2*M
      IRST=KSFT*L+K+1
120   IZZ(IRST)=JZZ
      RETURN
      END
```

```
      SUBROUTINE LVLSET
C
C  Purpose: Establish level-triplet to byte-packed conversion table
C               Levels (resolution codes) are concatenated as a
C                 three-digit octal number to form the look-up address,
C                 L3L2L1.  Maximum value is octal 555 for 32x32 case
C
C  CALLed from: SETUPM
C
C------------------------------------------------------------------------
      COMMON /LVLTBL/ LVLT(366)
      DIMENSION M(3,6)
      DATA LVLT/366*-1/
      DATA M/1,8,64, 1,64,8, 8,1,64, 64,1,8, 8,64,1, 64,8,1/
      LI=2
      DO 140 I1=1,6
      L1=I1-1
          DO 130 I2=1,I1
          L2=I2-1
              DO 120 I3=1,I2
              L3=I3-1
                  DO 110 IP=1,6
                  LV=L1*M(1,IP)+L2*M(2,IP)+L3*M(3,IP)+1
                  IF(LVLT(LV).NE.-1) GO TO 110
                  LI=LI+1
                  LVLT(LV)=LI
110               CONTINUE
120           CONTINUE
130       CONTINUE
140   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SFMAC(LOOPSF)
C
C  Purpose: Convert the subframe index, LOOPSF, to the corresponding
C              macro-fidelity control
C                Determine the center line and pixel of the subframe
C                Scale to the macro-fidelity coordinate indices
C                Generate the contrast control matrix base address for
C                  the corresponding fidelity control group (1-4)
C
C  CALLed from: MAPS
C
C-----------------------------------------------------------------------
      INTEGER*4 LOOPSF,ISTAR4
      COMMON /CONTRL/ MAC(16,16),KCS(4),CS(4),TP(4),SF(4),SB(4),
     +  KCM(4,5,4),LBP(6),KBP(2,9),IBP(2,9),IKSET
      COMMON /SFCNTL/ KSFT,KSQT,NPST,NPSL(8),MPSL(8),KSFH,FL,FP,JGRD
      COMMON /CNTRST/ KCMT(80),KINDX
      ISTAR4=(LOOPSF-1)/NPST                                    !  I*4
      NROW=ISTAR4
      LEFT=LOOPSF-NPST*ISTAR4                                   !  I*4
      IF(JGRD.NE.0) GO TO 110
C
C     Square Grid Partition
C
      K=KSFT*(LEFT-1)+KSFH
      L=KSFT*NROW+KSFH
      GO TO 150
C
C     Staggered Grid Partition
C
110   LFT=LEFT
      DO 120 J=1,8
      LTMP=LFT-NPSL(J)
      IF(LTMP.LE.0) GO TO 130
120   LFT=LTMP
130   K=KSQT*(LFT-1)+MPSL(J)
      L=KSFT*(NROW-1)+KSFH+J
150   MK=FP*K+1
      IF(MK.LT.1) MK=1
      IF(MK.GT.16) MK=16
      ML=FL*L+1
      IF(ML.LT.1) ML=1
      IF(ML.GT.16) ML=16
      KINDX=20*MAC(MK,ML)-19
      RETURN
      END
```

```
      SUBROUTINE SFIN
C
C  Purpose: Input and convert next source image subframe
C              Re-order from subframe raster to zigzag position
C              Remap gray scales from code to contrast and intensity
C                spaces
C              Initialize level and pattern assignments
C
C  CALLed from: MAPS
C
C-----------------------------------------------------------------
      COMMON /SFDATA/ ISF(1024)
      BYTE ISF
      COMMON /GRYSCL/ KRMP(256),IRMP(256),IDMP(4096)
      COMMON /ZIGZAG/ IZZ(1024)
      COMMON /MAPSSF/ NSQ,NLVL,ICODE(1024),KNTRST(1024),INTENS(1024),
     +  LEVEL(1024),PATTRN(1024)
      BYTE LEVEL,PATTRN
      READ (2) (ISF(J),J=1,NSQ)
      DO 110 J=1,NSQ
      N=ISF(J)
      IF(N.LT.0) N=N+256
      JZZ=IZZ(J)
      ICODE(JZZ)=N
      N=N+1
      K=KRMP(N)
      KNTRST(JZZ)=K
      I=IRMP(N)
      INTENS(JZZ)=I
      LEVEL(JZZ)=0
      PATTRN(JZZ)=0
110   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SFMAPS
C
C  Purpose: MAPS compression of source image subframe
C               Loop on levels
C                   Loop on MAPSel quads (in zigzag order)
C                       Form contrasts and sign-sort vector
C                       Test contrasts and form MAPSel (if required)
C               Recursion on completed MAPSels
C                   Accumulate optimum bias and performance statistics
C                   Transfer MAPSels to output buffer
C                   Output 512-byte MAPSel records asynchronously
C
C  CALLed from: MAPS
C
C  CALLs: QDIFF,THRESH,MAPOUT
C
C------------------------------------------------------------------------
      COMMON /MDATA/ MREC,MLOC,MSF(512)
      BYTE MSF
      INTEGER*4 MREC
      COMMON /MTEMP/ NN,KM,MAPSEL(5),MLVL(4)
      COMMON /CNTRST/ KCNT(80),KINDX
      COMMON /BLKPAT/ LBPT(5),IOBP(4,4)
      COMMON /GRYSCL/ KRMP(256),IRMP(256),IDMP(4096)
      COMMON /MAPSSF/ NSQ,NLVL,ICODE(1024),KNTRST(1024),INTENS(1024),
     +  LEVEL(1024),PATTRN(1024)
      BYTE LEVEL,PATTRN
      COMMON /MSTATS/ KOUNT(2,6),DIFF(2,6),DIFFSQ,SUMSQ
      INTEGER*4 KOUNT
      REAL*8 DIFF,DIFFSQ,SUMSQ
      COMMON /QUAD/ KT(4),IT(4),NT(6),LQ,LVLP,NN
      DIMENSION MSTEP(6)
      INTEGER*4 ISTAR4
      DATA MSTEP/1,4,16,64,256,1024/
      NTRNS=NLVL-1
      DO 130 LTRNS=1,NTRNS
      LSTP=MSTEP(LTRNS+1)
      MSTP=MSTEP(LTRNS)
      LVL=LTRNS-1
      LVLP=LTRNS
          DO 120 LQUAD=1,NSQ,LSTP
          LQ=LQUAD
          IMAPS=LQUAD
              DO 110 MPSEL=1,4
              IF(LEVEL(IMAPS).NE.LVL) GO TO 120
              KT(MPSEL)=KNTRST(IMAPS)
              IT(MPSEL)=INTENS(IMAPS)
110           IMAPS=IMAPS+MSTP
          CALL QDIFF
          CALL THRESH
120       CONTINUE
130   KINDX=KINDX+4
```

```
      LQ=1
200   LVL=LEVEL(LQ)
      LVLP=LVL+1
      INT=INTENS(LQ)+1
      KODE=IDMP(INT)
      IF(LVL.GT.0) GO TO 210
      IC=ICODE(LQ)
      ISTAR4=IC
      SUMSQ=SUMSQ+ISTAR4*ISTAR4                              !  I*4
      IM=IC-KODE
      ISTAR4=IM
      DIFFSQ=DIFFSQ+ISTAR4*ISTAR4                            !  I*4
      KOUNT(1,1)=KOUNT(1,1)+1                                !  I*4
      LQ=LQ+1
      KBPT=0
      GO TO 300
210   MSTP=MSTEP(LVL)
      IPAT=PATTRN(LQ)
      JPAT=1
      KBPT=LBPT(LVL)
      IF(KBPT.NE.0) KODE=KODE.AND."374
      DO 230 JQUAD=1,4
      IF(KBPT.NE.0) JPAT=IQBP(JQUAD,IPAT+1)
          DO 220 INQUAD=1,MSTP
          IC=ICODE(LQ)
          ISTAR4=IC
          SUMSQ=SUMSQ+ISTAR4*ISTAR4                          !  I*4
          IM=IC-KODE
          ISTAR4=IM
          DIFFSQ=DIFFSQ+ISTAR4*ISTAR4                        !  I*4
          DIFF(JPAT,LVLP)=DIFF(JPAT,LVLP)+KBPT*IM
          KOUNT(JPAT,LVLP)=KOUNT(JPAT,LVLP)+1                !  I*4
220       LQ=LQ+1
230   CONTINUE
300   KM=KM+1
      IF(KBPT.NE.0) KODE=KODE.OR.IPAT
      MAPSEL(KM+1)=KODE
      MLVL(KM)=LVL
      IF(KM.LT.NM) GO TO 310
      CALL MAPOUT
310   IF(LQ.LE.NSQ) GO TO 200
      RETURN
      END
```

```
      SUBROUTINE QDIFF
C
C  Purpose: Generate quad contrasts and sign-sort vector
C               Form all six quad contrasts: 3-2 3-1 2-0 1-0 2-1 3-0
C               Pack signs of contrasts in left-right order shown
C                 for sign-sort vector
C                   Lowest two bits automatically give pattern code
C
C  CALLed from: SFMAPS
C
C-----------------------------------------------------------------
      COMMON /QUAD/ KT(4),IT(4),NT(6),LQ,LVLP,NN
      NT(1)=KT(4)-KT(3)
      NT(2)=KT(4)-KT(2)
      NT(3)=KT(3)-KT(1)
      NT(4)=KT(2)-KT(1)
      NT(5)=KT(3)-KT(2)
      NT(6)=KT(4)-KT(1)
      NN=0
      DO 110 J=1,6
      NN=2*NN
      IF(NT(J).LT.0) NN=NN+1
110   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE THRESH
C
C  Purpose: Test contrasts and form new composite MAPSel (if required)
C              Based on sign-sort vector index NN:
C                  NSRT gives contrast indices in following order:
C                      (Extreme  Middle step  Lower step  Upper step)
C                  ISRT gives intensity indices in increasing-value order
C              Demap new MAPSel from intensity space; then remap to
C                contrast space
C
C  CALLed from: SFMAPS
C
C-----------------------------------------------------------------
      COMMON /CNTRST/ KCMT(80),KINDX
      COMMON /GRYSCL/ KRMP(256),IRMP(256),IDMP(4096)
      COMMON /RESET/ MSK(4),NORM,NBIAS
      COMMON /MAPSSF/ NSQ,NLVL,ICODE(1024),KNTRST(1024),INTENS(1024),
     +  LEVEL(1024),PATTRN(1024)
      BYTE LEVEL,PATTRN
      COMMON /QUAD/ KT(4),IT(4),NT(6),LQ,LVLP,NN
      DIMENSION NSRT(4,64),ISRT(4,64)
      DATA NSRT/6,5,4,1, 4*0, 6,5,3,2, 4*0, 2,3,4,1, 20*0, 1,4,3,2,
     +  4*0, 2,3,5,6, 4,1,5,6, 1,4,5,6, 3,2,5,6, 8*0, 4,1,3,2, 28*0,
     +  5,6,3,2, 5,6,1,4, 12*0, 3,2,1,4, 3,2,4,1, 12*0, 5,6,4,1,
     +  5,6,2,3, 28*0, 4,1,2,3, 8*0, 3,2,6,5, 1,4,6,5, 4,1,6,5,
     +  2,3,6,5, 4*0, 1,4,2,3, 20*0, 6,5,2,3, 4*0, 2,3,1,4, 4*0,
     +  6,5,1,4/
      DATA ISRT/1,2,3,4, 4*0, 1,3,2,4, 4*0, 2,1,3,4, 20*0, 3,1,2,4,
     +  4*0, 2,3,1,4, 2,3,4,1, 3,2,1,4, 3,2,4,1, 8*0, 1,3,4,2, 28*0,
     +  3,1,4,2, 3,4,1,2, 12*0, 3,4,2,1, 1,2,4,3, 12*0, 2,1,4,3,
     +  2,4,1,3, 28*0, 2,4,3,1, 8*0, 1,4,2,3, 4,1,2,3, 1,4,3,2,
     +  4,1,3,2, 4*0, 4,2,1,3, 20*0, 4,2,3,1, 4*0, 4,3,1,2, 4*0,
     +  4,3,2,1/
      NNP=NN+1
      KX=KINDX-1
      DO 110 J=1,4
      NDX=NSRT(J,NNP)
      KX=KX+1
      IF(IIABS(NT(NDX)).GT.KCMT(KX)) GO TO 130
110   CONTINUE
      NEW=NBIAS
      DO 120 J=1,4
      IDX=ISRT(J,NNP)
120   NEW=NEW+(MSK(J).AND.IT(IDX))
      NEW=NEW/NORM
      KNEW=IDMP(NEW+1)+1
      KNTRST(LQ)=KRMP(KNEW)
      INTENS(LQ)=NEW
      LEVEL(LQ)=LVLP
      PATTRN(LQ)=NN.AND."3
130   RETURN
      END
```

```
      SUBROUTINE MAPOUT
C
C  Purpose: Transfer completed MAPSel multiplets to output buffer
C           Pack level multiplets to byte (4 if 8x8, 3 if 16x16 32x32)
C           Transfer intensity/pattern codes to following bytes (4 3)
C           Output 512-byte MAPSel stream records asynchronously
C
C  CALLed from: SFMAPS
C
C--------------------------------------------------------------------
      COMMON /MDATA/ MREC,MLOC,MSF(512)
      BYTE MSF
      INTEGER*4 MREC
      COMMON /MTEMP/ NM,KM,MAPSEL(5),MLVL(4)
      COMMON /LVLTBL/ LVLT(366)
      IF(NM.EQ.3) GO TO 120
      M=1
      LV=0
      DO 110 J=1,KM
      LV=LV+M*MLVL(J)
110   M=4*M
      MAPSEL(1)=LV
      GO TO 140
120   M=1
      LV=1
      DO 130 J=1,KM
      LV=LV+M*MLVL(J)
130   M=8*M
      MAPSEL(1)=LVLT(LV)
140   KM=KM+1
      DO 150 J=1,KM
      MT=MAPSEL(J)
      IF(MT.GT.127) MT=MT-256
      MLOC=MLOC+1
      MSF(MLOC)=MT
      IF(MLOC.LT.512) GO TO 150
      MREC=MREC+1                                     ! I*4
      WRITE (3'MREC) MSF
      MLOC=0
150   CONTINUE
      KM=0
      RETURN
      END
```

```
      SUBROUTINE LSTREC
C
C  Purpose: Transfer remaining MAPSels to output buffer and complete
C           MAPSel stream
C
C  CALLed from: MAPS
C
C-----------------------------------------------------------------
      COMMON /MDATA/ MREC,MLOC,MSF(512)
      BYTE MSF
      INTEGER*4 MREC
      COMMON /MTEMP/ NM,KM,MAPSEL(5),MLVL(4)
      COMMON /LVLTBL/ LVLT(366)
      IF(KM.LE.0) GO TO 200
      IF(NM.EQ.3) GO TO 120
      M=1
      LV=0
      DO 110 J=1,KM
      LV=LV+M*MLVL(J)
110   M=4*M
      MAPSEL(1)=LV
      GO TO 140
120   M=1
      LV=1
      DO 130 J=1,KM
      LV=LV+M*MLVL(J)
130   M=8*M
      MAPSEL(1)=LVLT(LV)
140   KM=KM+1
      DO 150 J=1,KM
      MT=MAPSEL(J)
      IF(MT.GT.127) MT=MT-256
      MLOC=MLOC+1
      MSF(MLOC)=MT
      IF(MLOC.LT.512) GO TO 150
      MREC=MREC+1                                       !  I*4
      WRITE (3'MREC) MSF
      MLOC=0
150   CONTINUE
200   IF(MLOC.LE.0) GO TO 300
      MLP=MLOC+1
      DO 210 J=MLP,512
210   MSF(J)=0
      MREC=MREC+1                                       !  I*4
      WRITE (3'MREC) MSF
300   RETURN
      END
```

```
      SUBROUTINE SUMMRY
C
C  Purpose: Determine optimum pattern biases where desginated and
C           report overall compression and fidelity performance
C
C  CALLed from: MAPS
C
C-----------------------------------------------------------------
      COMMON /HEADER/ IFILE,INAME(8),NL,NP,NB,KSF,IGRD,NS,MC,MIXBP,
     +  IBV(5,2),IPAD(7)
      BYTE INAME
      INTEGER*4 NS,MC
      COMMON /MDATA/ MREC,MLOC,MSF(512)
      BYTE MSF
      INTEGER*4 MREC
      COMMON /MSTATS/ KOUNT(2,6),DIFF(2,6),DIFFSQ,SUMSQ
      INTEGER*4 KOUNT
      REAL*8 DIFF,DIFFSQ,SUMSQ
      DIMENSION MSIZE(5)
      INTEGER*4 LBIT,ISTAR4
      REAL*8 TEMP8,PIX,BITI,BITM
      DATA MSIZE/4,16,64,256,1024/
      MC=KOUNT(1,1)+KOUNT(2,1)                              !  I*4
      KOUNT(1,1)=MC
      DO 120 J=1,5
      JJ=J+1
          DO 110 I=1,2
          IOPT=0
          IF(KOUNT(I,JJ).EQ.0) GO TO 110
          TEMP8=KOUNT(I,JJ)
          OPT=DIFF(I,JJ)/TEMP8
          IOPT=OPT+0.5
          IF(OPT.LT.0.) IOPT=OPT-0.5
          DIFFSQ=DIFFSQ-2.*IOPT*DIFF(I,JJ)+IOPT*IOPT*TEMP8
110       IBV(J,I)=IOPT
      KOUNT(1,JJ)=(KOUNT(1,JJ)+KOUNT(2,JJ))/MSIZE(J)        !  I*4
      MC=MC+KOUNT(1,JJ)                                     !  I*4
120   CONTINUE
      TYPE 500,MREC-2,MLOC
500   FORMAT(/,1X,'MAPS FILE CONTAINS',I6,' 512-BYTE RECORDS PLUS',
     +  I4,' BYTES IN THE LAST')
      NLVL=4
      IF(KSF.EQ.16) NLVL=5
      IF(KSF.EQ.32) NLVL=6
      TYPE 510,(L-1,L=1,NLVL)
510   FORMAT(/,1X,'MAPSEL DISTRIBUTION:',//,3X,'LEVEL:';I7,5I9)
      TYPE 520,(KOUNT(1,L),L=1,NLVL)
520   FORMAT(3X,'COUNT:',6I9)
      NLVL=NLVL-1
      TYPE 530,(IBV(J,1),J=1,NLVL)
530   FORMAT(/,3X,'OPTIMAL BIAS: -',I7,4I9)
      TYPE 540,(IBV(J,2),J=1,NLVL)
540   FORMAT(17X,'+',I7,4I9)
```

```
      ISTAR4=NL
      PIX=NP*ISTAR4                                          !  I*4
      BIT1=NB*PIX
      LBIT=(MC+2)/3                                          !  I*4
      IF(KSF.EQ.8) LBIT=(MC+3)/4                             !  I*4
      BITM=8*(MC+LBIT)                                       !  I*4
      CR=BIT1/BITM
      BPP=BITM/PIX
      ERROR=100.*(DIFFSQ/SUMSQ)
      TYPE 550,CR,BPP,ERROR
550   FORMAT(/,1X,'COMPRESSION RATIO:',F8.3,' : 1',//,1X,
     +   'BITS/PIXEL:',F8.5,//,1X,'MEAN SQUARE ERROR:',F9.5,' %')
      RETURN
      END
```

SECTION NINE

TRANSMAPS MODULE #3: MAPS DECOMPRESSION AND RESOLUTION IMAGE FORMATION

9.1 Program Characteristics

Program Names: DMAPS or DM

Subroutines:	USERD FILESD SETUPD GAZGIZ LDCSET SFDMAP	
Files:	MAPS.DAT DMAPS.DAT LEVEL.DAT	(input) (output) (output)

Task Build Options: MAXBUF = 1024

Task Size: 18720

9.2 Source Listing

The COMMENT-annotated source listing for DMAPS follows:

```
      PROGRAM DMAPS
C
C  +---------------------------------------------------------------+
C  |                                                               |
C  |  TransMAPS Module #3:  MAPS Decompression & Level Image Formation  |
C  |                                                               |
C  +---------------------------------------------------------------+
C
C                                     Control Data Corporation - 1982
C
C Files: Unit  Name    Content                        From/To  Type
C ------
C   In    2    MAPS    MAPSel stream, 512 byte rec.    MAPS    FIXED SEQ.
C   Out   3    DMAPS   MAPS Decompressed image,        ADAPT   FIXED SEQ.
C                        subframes                   or DIFFER
C                                                    or RASTER
C   Out   4    LEVEL   Level (Resolution) image,       ADAPT   FIXED SEQ.
C                        subframes                   or RASTER
C
C User Interaction:  No parameter inputs required
C -----------------
C
C Program Structure:
C ------------------
C       PROGRAM DMAPS
C           CALL USERD   Dummy routine for future user interaction
C           CALL FILESD   Open files and read/write headers
C           CALL SETUPD   Establish control tables
C               CALL GAZGIZ   Establish zigzag to raster conversion table
C               CALL LDCSET   Establish byte to level-multiplet table
C               Establish intensity/pattern decode table
C           Loop on subframes
C               CALL SFDMAP   Convert MAPSel stream to decompressed and
C                               resolution (level) images by subframe
C
C COMMON Block Communication:
C ---------------------------
C    /BLKPAT/   Block/pattern mode by level             Length: 21
C                   SETUPD,SFDMAP
C
C       LBPT(5)          [I*2]   Block(0)/pattern(1) for levels 1-5
C       IQBP(4,4)        [I*2]   Low/High index by Quadrant,Pattern
C
C    /DMAPSF/   Decompression and level image output    Length: 1025
C                   FILESD,SFDMAP
C
C       NSQ              [I*2]   Number of total pixels/subframe
C       ISF(1024)        [Byte]  Decompressed subframe array
C       LSF(1024)        [Byte]  Level image subframe array
C
C    /GAZGIZ/   Zigzag to raster conversion             Length: 1024
C                   GAZGIZ,SFDMAP
C
C       NZZ(1024)        [I*2]   Zigzag to raster conversion table
C
```

```
C     /HEADER/   Standard MAPS file header                 Length: 32
C                    DMAPS,FILESD,SETUPD
C
C        IFILE           [I*2]   File type
C        INAME(8)        [Byte]  User-selected image name
C        NL              [I*2]   Number of lines in source image
C        NP              [I*2]   Number of pixels in source image
C        NB              [I*2]   Number of bits/pixel in source image
C        KSF             [I*2]   Kind of subframe 8x8 16x16 32x32
C        IGRD            [I*2]   Subframe grid: square(0)/staggered(1)
C        NS              [I*4]   Total subframe count
C        MC              [I*4]   MAPSel count
C        MIXBP           [I*2]   Packed block(0)/pattern(1) mode (rt-lft)
C        IBV(5,2)        [I*2]   Optimal pattern biases by level,low/high
C        IPAD(7)         [I*2]   Space for future extension
C
C     /LDCODE/   Byte-packed Level decode                  Length: 1024
C                    LDCSET,SFDMAPS
C
C        LDC(4,256)      [I*2]   Byte to level-multiplet decoding table
C
C     /MDATA/    MAPSel stream data                        Length: 260
C                    FILESD,SFDMAP
C
C        MLOC            [I*2]   Current location in MAPS input buffer
C        MSF(512)        [Byte]  MAPSel stream input buffer
C        MLVL            [I*2]   Current packed level byte
C        NM              [I*2]   Number of levels/byte 8x8(4),
C                                  16x16 or 32x32(3)
C        KM              [I*2]   Current position of level in byte (1-NM)
C
C     /MDCODE/   Intensity/pattern decode                  Length: 2560
C                    SETUPD,SFDMAP
C
C        MDC(256,4,5)    [I*2]   Intensity/pattern decoding table by
C                                  intensity/pattern byte,quadrant,level
C
C------------------------------------------------------------------
      COMMON /HEADER/ IFILE,INAME(8),NL,NP,NB,KSF,IGRD,NS,MC,MIXBP,
     +  IBV(5,2),IPAD(7)
      BYTE INAME
      INTEGER*4 NS,MC
      INTEGER*4 LOOPSF
      CALL USERD
      CALL FILESD
      TYPE 500,INAME,NL,NP
500   FORMAT(/,1X,'MAPS DECOMPRESSING IMAGE ',8A1,',',I5,' LINES BY',
     +  I5,' PIXELS')
      CALL SETUPD
      DO 110 LOOPSF=1,NS                                    !  I*4
110   CALL SFDMAP
      CLOSE(UNIT=2)
      CLOSE(UNIT=3)
      CLOSE(UNIT=4)
      END
```

```
      SUBROUTINE USERD
C
C  Purpose: User interaction for MAPS decompression and level image
C              formation
C                Dummy routine for possible future extensions
C
C  CALLed from: DMAPS
C
C-----------------------------------------------------------------------
      TYPE 500
500   FORMAT(/,1X,46(1H*),/,1X,'*',44X,'*',/,1X,
     +   '* MAPS DECOMPRESSION/RESOLUTION IMAGE MODULE *',/,1X,'*',44X,
     +   '*',/,1X,46(1H*),//,3X,'NO USER INPUTS REQUIRED')
      RETURN
      END
```

```
      SUBROUTINE FILESD
C
C  Purpose: Open files, read/write standard MAPS headers
C
C  CALLed from: DMAPS
C
C-----------------------------------------------------------------
      COMMON /HEADER/ IFILE,INAME(8),NL,NP,NB,KSF,NS,MC,MIXBP,
     +  IBV(5,2),IPAD(7)
      BYTE INAME
      INTEGER*4 NS,MC
          DIMENSION JHEAD(32)
          EQUIVALENCE (JHEAD(1),IFILE)
      COMMON /MDATA/ MLOC,MSF(512),MLVL,NM,KM
      BYTE MSF
      COMMON /DMAPSF/ NSQ,ISF(1024),LSF(1024)
      BYTE ISF,LSF
           DIMENSION JSF(512)
           EQUIVALENCE (JSF(1),ISF(1))
      DATA ISF/1024*0/,LSF/1024*0/
      OPEN(UNIT=2,TYPE='OLD',NAME='MAPS',FORM='UNFORMATTED',
     +  RECORDTYPE='FIXED')
      READ (2) JHEAD
      IF(IFILE.EQ.1) GO TO 110
              TYPE 500,IFILE
500           FORMAT(/,1X,'*** FILE TYPE',I3,' NOT MAPSel STREAM')
              STOP
110   READ (2) MSF
      MLOC=1
      MLVL=MSF(1)
      IF(MLVL.LT.0) MLVL=MLVL+256
      MLVL=MLVL+1
      NM=4
      IF(KSF.NE.8) NM=3
      KM=0
      NSQ=KSF*KSF
      LSQ=NSQ/4
      DO 120 J=1,32
120   JSF(J)=JHEAD(J)
      OPEN(UNIT=3,TYPE='NEW',NAME='DMAPS',FORM='UNFORMATTED',
     +  RECORDTYPE='FIXED',RECORDSIZE=LSQ)
      JSF(1)=2
      WRITE (3) (ISF(J),J=1,NSQ)
      OPEN(UNIT=4,TYPE='NEW',NAME='LEVEL',FORM='UNFORMATTED',
     +  RECORDTYPE='FIXED',RECORDSIZE=LSQ)
      JSF(1)=3
      WRITE (4) (ISF(J),J=1,NSQ)
      DO 130 J=1,32
130   JSF(J)=0
      RETURN
      END
```

```
      SUBROUTINE SETUPD
C
C  Purpose: Establish decompression control tables
C              Establish zigzag to subframe raster conversion table
C              Establish byte to level multiple  resolution decode table
C              Establish intensity/pattern byte decode table
C                    by byte value, quadrant, level
C
C  CALLed from: DMAPS
C
C  CALLs: GAZGIZ,LDCSET
C
C-----------------------------------------------------------------------
      COMMON /HEADER/ IFILE,INAME(8),NL,NP,NB,KSF,IGRD,NS,MC,MIXBP,
     +   IBV(5,2),IPAD(7)
      BYTE INAME
      INTEGER*4 NS,MC
      COMMON /MDCODE/ MDC(256,4,5)
      BYTE MDC
      COMMON /BLKPAT/ LBPT(5),IQBP(4,4)
      DATA IQBP/1,1,2,2, 2,1,2,1, 1,2,1,2, 2,2,1,1/
      KSFT=KSF
      CALL GAZGIZ(KSFT)
      CALL LDCSET(KSFT)
      M=2
      DO 110 J=1,5
      LBPT(J)=(MIXBP/M).AND."1
110   M=2*M
      DO 150 LV=1,5
      KBPT=LBPT(LV)
         DO 140 JQ=1,4
            DO 130 I=1,256
            M=I-1
            IF(KBPT.EQ.0) GO TO 120
            IPAT=M.AND."3
            INT=M-IPAT
            IBP=IQBP(JQ,IPAT+1)
            JBV=IBV(LV,IBP)
            M=INT+JBV
            IF(M.LT.0) M=0
            IF(M.GT.255) M=255
120         IF(M.GT.127) M=M-256
130         MDC(I,JQ,LV)=M
140      CONTINUE
150   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE GAZGIZ(KSFT)
C
C  Purpose: Establish zigzag to subframe raster conversion table
C
C  CALLed from: SETUPD
C
C-----------------------------------------------------------------------
      COMMON /GAZGIZ/ NZZ(1024)
      DATA NZZ/1024*0/
      KSQT=KSFT*KSFT
      DO 120 JZZ=1,KSQT
      NZZ=JZZ-1
      KMSK=NZZ.AND."525
      LMSK=(NZZ/2).AND."525
      K=0
      L=0
      M=1
          DO 110 J=1,5
          KT=KMSK.AND."1
          K=K.OR.(M*KT)
          LT=LMSK.AND."1
          L=L.OR.(M*LT)
          KMSK=KMSK/4
          LMSK=LMSK/4
110       M=2*M
      IRST=KSFT*L+K+1
120   NZZ(JZZ)=IRST
      RETURN
      END
```

```
      SUBROUTINE LDCSET(KSFT)
C
C  Purpose: Establish byte to level multiplet resolution decode table
C             by MAPSel sequence (1-4 for 8x8, 1-3 for 16x16 & 32x32)
C             and byte value
C
C  CALLed from: SETUPD
C
C---------------------------------------------------------------------
      COMMON /LDCODE/ LDC(4,256)
      DIMENSION M(3,6),LVLT(366)
      DATA LDC/1024*-1/
      DATA LVLT/366*-1/
      DATA M/1,8,64, 1,64,8, 8,1,64, 64,1,8, 8,64,1, 64,8,1/
      IF(KSFT.NE.8) GO TO 200
      DO 120 I=1,256
      LPACK=I-1
      MT=1
          DO 110 J=1,4
          LDC(J,I)=(LPACK/MT).AND."3
110       MT=4*MT
120   CONTINUE
      GO TO 300
200   LI=2
      DO 240 I1=1,6
      L1=I1-1
          DO 230 I2=1,I1
          L2=I2-1
              DO 220 I3=1,I2
              L3=I3-1
                  DO 210 IP=1,6
                  LV=L1*M(1,IP)+L2*M(2,IP)+L3*M(3,IP)+1
                  IF(LVLT(LV).NE.-1) GO TO 210
                  LI=LI+1
                  LVLT(LV)=LI
210               CONTINUE
220           CONTINUE
230       CONTINUE
240   CONTINUE
      DO 260 IT=1,366
      I=LVLT(IT)
      IF(I.EQ.-1) GO TO 260
      LPACK=IT-1
      MT=1
          DO 250 J=1,3
          LDC(J,I+1)=(LPACK/MT).AND."7
250       MT=8*MT
260   CONTINUE
300   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SFDMAP
C
C  Purpose: Convert MAPSel stream to decompressed and resolution (level)
C            images by subframe
C              Recursion on MAPSels (automatic zigzag order)
C                  Retrieve level code
C                  Retrieve intensity/pattern byte
C                      Asynchronously input next MAPS record as needed
C                  Loop on MAPSel quadrants
C                      Convert intensity/pattern byte
C                      Loop on pixels within quadrant in zigzag order
C                          Determine subframe raster address
C                          Set decompressed and level image pixel
C                  Return for next MAPSel until subframe completed
C              Output decompressed and level image subframes
C
C  CALLed from: DMAPS
C
C-----------------------------------------------------------------------
      COMMON /MDATA/ MLOC,MSF(512),MLVL,NM,KM
      BYTE MSF
      COMMON /DMAPSF/ NSQ,ISF(1024),LSF(1024)
      BYTE ISF,LSF
      COMMON /GAZGIZ/ NZZ(1024)
      COMMON /LDCODE/ LDC(4,256)
      COMMON /MDCODE/ MDC(256,4,5)
      BYTE MDC
      COMMON /BLKPAT/ LBPT(5),IQBP(4,4)
      DIMENSION MSTEP(5)
      DATA MSTEP/1,4,16,64,256/
      JZZ=0
100   KM=KM+1
      IF(KM.LE.NM) GO TO 120
      KM=1
      MLOC=MLOC+1
      IF(MLOC.LE.512) GO TO 110
      MLOC=1
      READ (2) MSF
110   MLVL=MSF(MLOC)
      IF(MLVL.LT.0) MLVL=MLVL+256
      MLVL=MLVL+1
120   LV=LDC(KM,MLVL)
      MLOC=MLOC+1
      IF(MLOC.LE.512) GO TO 130
      MLOC=1
      READ (2) MSF
130   MI=MSF(MLOC)
      IF(LV.GT.0) GO TO 200
      JZZ=JZZ+1
      IRST=NZZ(JZZ)
      ISF(IRST)=MI
      LSF(IRST)=0
      IF(JZZ.GE.NSQ) GO TO 300
      GO TO 100
```

```
200     MSTP=MSTEP(LV)
        IPAT=0
        IF(LBPT(LV).NE.0) IPAT=MI.AND."3
        L=32*LV+4*IPAT
        IF(L.GT.127) L=L-256
        IF(MI.LT.0) MI=MI+256
        MI=MI+1
        DO 220 JQ=1,4
        I=MDC(MI,JQ,LV)
            DO 210 J=1,MSTP
            JZZ=JZZ+1
            IRST=NZZ(JZZ)
            ISF(IRST)=I
210         LSF(IRST)=L
220     CONTINUE
        IF(JZZ.LT.NSQ) GO TO 100
300     WRITE (3) (ISF(J),J=1,NSQ)
        WRITE (4) (LSF(J),J=1,NSQ)
        RETURN
        END
```

SECTION TEN

TRANSMAPS MODULE #4: MAPS ADAPTIVE IMAGE SMOOTHING

10.1 Program Characteristics

Program Names:	ADAPT or AD	
Subroutines:	USERA FILESA SETUPA LINPIX WGTSET UNIFRM GAUSS SFLOAD SFUPDT SFADPT	
Files:	DMAPS.DAT LEVEL.DAT ADAPT.DAT	(input) (input) (output)
Task Build Options:	MAXBUF = 1024	
Task Size:	27968	

10.2 Source Listing

The COMMENT-annotated source listing for ADAPT follows:

```
      PROGRAM ADAPT
C
C   +-----------------------------------------------------+
C   |                                                     |
C   |  TransMAPS Module #4:  MAPS Adaptive Image Smoothing |
C   |                                                     |
C   +-----------------------------------------------------+
C
C                                     Control Data Corporation - 1982
C
C Files: Unit  Name    Content                        From/To  Type
C ------
C   In    2    DMAPS   MAPS Decompressed image,        DMAPS   DIRECT
C                        subframes
C   In    3    LEVEL   Level (Resolution) image,       DMAPS   DIRECT
C                        subframes
C   Out   4    ADAPT   Adaptively smoothed image,      DIFFER  FIXED SEQ.
C                        subframes                     RASTER
C
C User Interaction:  In Subroutine USERA
C -----------------
C       Convolution Weighting Specification
C       Dither Amplitude Specification
C
C Program Structure:
C ------------------
C       PROGRAM ADAPT
C           CALL USERA   Specify convolution weighting and dither
C           CALL FILESA   Open files, check integrity, read/write headers
C           CALL SETUPA   Establish control tables, subframe partition
C               CALL LINPIX   Establish zigzag to line, pixel, & raster
C                               index conversion tables
C               CALL WGTSET   Establish pre-summed convolution weights
C                   CALL UNIFRM(LVL)   Establish uniform window weights
C                 or CALL GAUSS(LVL)    Establish Gaussian window weights
C           Loop on subframe rows in line direction
C               CALL SFLOAD   Initialize first subframe in row and its
C                               surround
C               Loop on subframes within row
C                   CALL SFUPDT   Update to next subframe & its surround
C                   CALL SFADPT   Adaptively smooth MAPSels in subframe
C
C COMMON Block Communication:
C ---------------------------
C    /GAZGIZ/   Zigzag to raster conversion             Length: 1024
C                   LINPIX,SFADPT
C
C       NZZ(1024)       [I*2]   Zigzag to raster conversion table
C
```

```
C     /HEADD/     Standard MAPS file header from DMAPS     Length: 32
C                     ADAPT,FILESA,SETUPA
C
C        IFILED           [I*2]    File type
C        INAMED(8)        [Byte]   User-selected image name
C        NLD              [I*2]    Number of lines in source image
C        NPD              [I*2]    Number of pixels in source image
C        NBD              [I*2]    Number of bits/pixel in source image
C        KSFD             [I*2]    Kind of subframe 8x8 16x16 32x32
C        IGRDD            [I*2]    Subframe grid: square(0)/staggered(1)
C        NSD              [I*4]    Total subframe count
C        MCD              [I*4]    MAPSel count
C        MIXBPD           [I*2]    Packed block(0)/pattern(1) mode (rt-lft)
C        IBVD(5,2)        [I*2]    Optimal pattern biases by level,low/high
C        IPADD(7)         [I*2]    Space for future extension
C
C     /HEADL/     Standard MAPS file header from LEVEL     Length: 32
C                     FILESA
C
C        IFILEL           [I*2]    File type
C        INAMEL(8)        [Byte]   User-selected image name
C        NLL              [I*2]    Number of lines in source image
C        NPL              [I*2]    Number of pixels in source image
C        NBL              [I*2]    Number of bits/pixel in source image
C        KSFL             [I*2]    Kind of subframe 8x8 16x16 32x32
C        IGRDL            [I*2]    Subframe grid: square(0)/staggered(1)
C        NSL              [I*4]    Total subframe count
C        MCL              [I*4]    MAPSel count
C        MIXBPL           [I*2]    Packed block(0)/pattern(1) mode (rt-lft)
C        IBVL(5,2)        [I*2]    Optimal pattern biases by level,low/high
C        IPADL(7)         [I*2]    Space for future extension
C
C     /LPZZ/      Zigzag to line/pixel conversion          Length: 2048
C                     LINPIX,WGTSET,SFADPT
C
C        LZZ(1024)        [I*2]    Zigzag to line conversion table
C        KZZ(1024)        [I*2]    Zigzag to pixel conversion table
C
C     /SFDATA/    Adaptive image subframe data             Length: 512
C                     FILESA,SFADPT
C
C        ISFA(1024)       [Byte]   Adaptive image subframe array
C
C     /SFINPT/    Input image (DMAPS,LEVEL) control        Length: 12
C                     SETUPA,SFLOAD,SFUPDT
C
C        K1ST             [I*2]    First pixel in buffer
C        KLST             [I*2]    Last pixel in buffer
C        L1ST             [I*2]    First line in buffer
C        LLST             [I*2]    Last line in buffer
C        KREC             [I*4]    Record number of current subframe
C        KBCK             [I*4]    Record number of adjacent prior
C                                    row subframe
C        KFWD             [I*4]    Record number of adjacent following
C                                    row subframe
C        NREC             [I*4]    Total number of input file records
C
```

```
C    /SFLOOP/   Subframe control parameters         Length: 6
C                   ADAPT,SETUPA,LINPIX,SFUPDT,SFADPT
C
C       KSF             [I*2]   Subframe size: edge
C       NSQ             [I*2]   Subframe size: number of total pixels
C       NLS             [I*2]   Number of subframes in line direction
C       NPS             [I*2]   Number of subframes in pixel direction
C       KLS             [I*2]   Current subframe index in line direction
C       KPS             [I*2]   Current subframe index in pixel direction
C
C    /SHARE/    DMAPS/LEVEL data; shared with weight    Length: 2210
C                 window during weight table generation
C                   WGTSET,UNIFRM,GAUSS,
C                   SFLOAD,SFUPDT,SFADPT
C
C       ISFD(65,34)     [Byte]  DMAPS subframe and surround data array
C       ISFL(65,34)     [Byte]  LEVEL subframe and surround data array
C         Equivalenced with:
C       WNDW(31,31)     [R*4]   Convolution weight window
C
C    /WGTDIT/   Convolution and dither selections       Length: 5
C                   USERA,WGTSET,GAUSS,SFADPT
C
C       KCW             [I*2]   Kind of convolution weight:
C                                 Uniform(U) or Gaussian(G)
C       SIG             [R*4]   Multiple of Gaussian sigma at
C                                 window corner
C       DIT             [R*4]   Dither amplitude
C
C    /WGTTBL/   Pre-summed convolution weights          Length: 6120
C                   WGTSET,SFADPT
C
C       WGT(9,340)      [R*4]   Weight table addressed by:
C                                 surround index,
C                                 compound level/quadrant-zigzag index
C
C MAPSel Surround Conventions:
C ----------------------------
C                 --->
C            Pixel direction
C                                Target  MAPSel         Weight Index
C       L      |        |       Region Quadrant 1  2  3  4  5  6  7  8  9
C       i    1 |  2  3  | 4                     Associated Surround Index
C       n      |        |
C       e  ----+--------+----      6       0    1  2  3  5  6  7  9 10 11
C              |        |
C    |  D    5 |  6  7  | 8        7       1    4  3  2  8  7  6 12 11 10
C    |  i      |        |
C    V  r    9 | 10  11 | 12      10       2   13 14 15  9 10 11  5  6  7
C       e      |        |
C       c  ----+--------+----     11       3   16 15 14 12 11 10  8  7  6
C       t      |        |
C       i   13 | 14  15 | 16
C       o      |        |
C       n
C
C------------------------------------------------------------------------
```

```
      COMMON /HEADD/ IFILED,INAMED(8),NLD,NPD,NBD,KSFD,IGRDD,NSD,MCD,
     +  MIXBPD,IBVD(5,2),IPADD(7)
      BYTE INAMED
      INTEGER*4 NSD,MCD
      COMMON /SFLOOP/ KSF,NSQ,NLS,NPS,KLS,KPS
      CALL USERA
      CALL FILESA
      TYPE 500,INAMED,NLD,NPD
500   FORMAT(/,1X,'MAPS ADAPTIVE SMOOTHING IMAGE ',8A1,',',I5,
     +  ' LINES BY',I5,' PIXELS')
      CALL SETUPA
      DO 120 ILS=1,NLS
      KLS=ILS
      KPS=1
      CALL SFLOAD
          DO 110 IPS=1,NPS
          KPS=IPS
          CALL SFUPDT
          CALL SFADPT
110       CONTINUE
120   CONTINUE
      CLOSE(UNIT=2)
      CLOSE(UNIT=3)
      CLOSE(UNIT=4)
      END
```

```
      SUBROUTINE USERA
C
C  Purpose: User interaction for MAPS adaptive image smoothing
C              Convolution weighting
C              Dither Amplitude
C
C  CALLed from: ADAPT
C
C---------------------------------------------------------------------------
      COMMON /WGTDIT/ KCW,SIG,DIT
      DATA KCW/1HG/,SIG/2.0/                                  ! Default
      DATA DIT/4.0/                                           ! Default
      TYPE 500
500   FORMAT(/,1X,34(1H*),/,1X,'*',32X,'*',/,1X,
     +   '* MAPS ADAPTIVE SMOOTHING MODULE *',
     +   /,1X,'*',32X,'*',/,1X,34(1H*))
100   TYPE 510,KCW
510   FORMAT(//,3X,'CONVOLUTION WEIGHTING:',//,5X,
     +   'UNIFORM OR GAUSSIAN? (U G)  ',A1)
      ACCEPT 1,LIT
1     FORMAT(10A1)
      IF(LIT.EQ.1HU) KCW=LIT
      IF(LIT.EQ.1HG) KCW=LIT
      IF(KCW.EQ.1HU) GO TO 110
      TYPE 520,SIG
520   FORMAT(5X,'SIGMA MULTIPLE AT WINDOW CORNER?',F5.1,2X,
     +   '(/ = NO CHNG)')
      ACCEPT *,SIG
110   TYPE 530,DIT
530   FORMAT(/,3X,'RANDOM DITHER:',//,5X,'AMPLITUDE?',F5.1,24X,
     +   '(/ = NO CHNG)')
      ACCEPT *,DIT
      TYPE 540
540   FORMAT(//,3X,'USER SPECIFICATION COMPLETE:',/,3X,28(1H*),//,5X,
     +   'REVIEW? (Y OR N)  N')
      ACCEPT 1,LIT
      IF(LIT.EQ.1HY) GO TO 100
      RETURN
      END
```

```
      SUBROUTINE FILESA
C
C  Purpose: Open files, check integrity, read/write headers
C
C  CALLed from: ADAPT
C
C-----------------------------------------------------------------------
      COMMON /HEADD/ IFILED,INAMED(8),NLD,NPD,NBD,KSFD,IGRDD,NSD,MCD,
     +  MIXBPD,IBVD(5,2),IPADD(7)
      BYTE INAMED
      INTEGER*4 NSD,MCD
          DIMENSION JHEADD(32)
          EQUIVALENCE (JHEADD(1),IFILED)
      COMMON /HEADL/ IFILEL,INAMEL(8),NLL,NPL,NBL,KSFL,IGRDL,NSL,MCL,
     +  MIXBPL,IBVL(5,2),IPADL(7)
      BYTE INAMEL
      INTEGER*4 NSL,MCL
          DIMENSION JHEADL(32)
          EQUIVALENCE (JHEADL(1),IFILEL)
      COMMON /SFDATA/ ISFA(1024)
      BYTE ISFA
          DIMENSION JSF(512)
          EQUIVALENCE (JSF(1),ISFA(1))
      OPEN(UNIT=2,TYPE='OLD',NAME='DMAPS',FORM='UNFORMATTED',
     +  RECORDTYPE='FIXED',ACCESS='DIRECT')
      READ (2'1) JHEADD
      IF(IGRDD.NE.0) GO TO 210
      OPEN(UNIT=3,TYPE='OLD',NAME='LEVEL',FORM='UNFORMATTED',
     +  RECORDTYPE='FIXED',ACCESS='DIRECT')
      READ (3'1) JHEADL
      IF(IGRDL.NE.0) GO TO 210
      DO 110 J=1,8
      IF(INAMED(J).NE.INAMEL(J)) GO TO 220
110   CONTINUE
      IF(NSD.NE.NSL) GO TO 220
      IF(MCD.NE.MCL) GO TO 220
      KSQT=KSFD*KSFD
      LSQ=KSQT/4
      OPEN(UNIT=4,TYPE='NEW',NAME='ADAPT',FORM='UNFORMATTED',
     +  RECORDTYPE='FIXED',RECORDSIZE=LSQ)
      DO 120 J=1,32
120   JSF(J)=JHEADD(J)
      JSF(1)=4
      WRITE (4) (ISFA(J),J=1,KSQT)
      GO TO 300
210           TYPE 510
510           FORMAT(/,1X,'*** NON-SQUARE GRID, ADAPT PRECLUDED')
              STOP
220           TYPE 520,INAMED,INAMEL,NDS,NSL,MCD,MCL
520           FORMAT(/,1X,'*** DMAPS AND LEVEL FILES UNMATCHED:',/,
     +          8X,'IMAGE: ',8A1,' VS ',8A1,/,8X,'SUBFRAMES:',I7,
     +          ' VS',I7,/,8X,'MAPSels:'I9,'VS',I9)
              STOP
300   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SETUPA
C
C  Purpose: Establish adaptive smoothing control tables
C             Establish inter-subframe position controls
C             Establish zigzag to line, pixel, and subframe raster
C               conversion tables
C             Establish pre-summed convolution weights
C
C  CALLed from: ADAPT
C
C  CALLs: LINPIX,WGTSET
C
C------------------------------------------------------------------
      COMMON /HEADD/ IFILED,INAMED(8),NLD,NPD,NBD,KSFD,IGRDD,NSD,MCD,
     +   MIXBPD,IBVD(5,2),IPADD(7)
      BYTE INAMED
      INTEGER*4 NSD,MCD
      COMMON /SFLOOP/ KSF,NSQ,NLS,NPS,KLS,KPS
      COMMON /SFINPT/ K1ST,KLST,L1ST,LLST,KREC,KBCK,KFWD,NREC
      INTEGER*4 KREC,KBCK,KFWD,NREC
      KSF=KSFD
      NSQ=KSF*KSF
      NLS=(NLD-1)/KSF+1
      NPS=(NPD-1)/KSF+1
      K1ST=KSF+2
      KLST=2*KSF+1
      L1ST=2
      LLST=KSF+1
      KREC=1
      KBCK=KREC-NPS
      KFWD=KREC+NPS
      NREC=NSD+1                                          !  I*4
      CALL LINPIX
      CALL WGTSET
      RETURN
      END
```

```
      SUBROUTINE LINPIX
C
C  Purpose: Establish zigzag to line, pixel, and subframe raster
C           index conversion tables
C
C  CALLed from: SETUPA
C
C-----------------------------------------------------------------
      COMMON /SFLOOP/ KSF,NSQ,NLS,NPS,KLS,KPS
      COMMON /GAZGIZ/ NZZ(1024)
      COMMON /LPZZ/ LZZ(1024),KZZ(1024)
      DATA NZZ/1024*0/
      KSFT=KSF
      DO 120 JZZ=1,1024
      NZZ=JZZ-1
      KMSK=NZZ.AND."525
      LMSK=(NZZ/2).AND."525
      K=0
      L=0
      M=1
         DO 110 J=1,5
         KT=KMSK.AND."1
         K=K.OR.(M*KT)
         LT=LMSK.AND."1
         L=L.OR.(M*LT)
         KMSK=KMSK/4
         LMSK=LMSK/4
110      M=2*M
      LZZ(JZZ)=L+2
      KZZ(JZZ)=K+2
      IRST=KSFT*L+K+1
120   NZZ(JZZ)=IRST
      RETURN
      END
```

```
      SUBROUTINE WGTSET
C
C  Purpose: Establish pre-summed convolution weights
C                Loop on levels of target MAPSel
C                    Establish convolution weight window:
C                      uniform or Gaussian
C                    Loop on pixels in first MAPSel quadrant; zigzag order
C                        Determine line and pixel ranges for 9 distinct
C                          MAPSel and surround regions
C                        Loop on regions
C                            Loop on lines, pixels within regions
C                            Sum convolution weights over region
C                        Set weight by region and pixel zigzag index
C
C  CALLed from: SETUPA
C
C  CALLs: UNIFRM,GAUSS
C
C-----------------------------------------------------------------------
      COMMON /WGTDIT/ KCW,SIG,DIT
      COMMON /SHARE/ ITEMP(2210)
      DIMENSION WNDW(31,31)
      EQUIVALENCE (WNDW(1,1),ITEMP(1))
      COMMON /LPZZ/ LZZ(1024),KZZ(1024)
      COMMON /WGTTBL/ WGT(9,340)
      DIMENSION LW1(3),LWL(3),KW1(3),KWL(3)
      DIMENSION MQSZ(5),LWC(5),LWNDW(5)
      DATA MQSZ/1,4,16,64,256/,LWC/1,2,4,8,16/,LWNDW/1,3,7,15,31/
      IW=0
      LW1(1)=1
      KW1(1)=1
      DO 160 LV=2,5
      LVL=LV
      IF(KCW.NE.1HG) CALL UNIFRM(LVL)
      IF(KCW.EQ.1HG) CALL GAUSS(LVL)
      NQ=MQSZ(LV)
      LWCT=LWC(LV)
      LWL(3)=LWNDW(LV)
      KWL(3)=LWNDW(LV)
          DO 150 JZZ=1,NQ
          IW=IW+1
          L=LZZ(JZZ)-1
          K=KZZ(JZZ)-1
          LWL(1)=LWCT-L
          LW1(2)=LWL(1)+1
          LWL(2)=LWL(1)+LWCT
          LW1(3)=LWL(2)+1
          KWL(1)=LWCT-K
          KW1(2)=KWL(1)+1
          KWL(2)=KWL(1)+LWCT
          KW1(3)=KWL(2)+1
          JW=0
```

```
                 DO 140 LREG=1,3
                 LW1ST=LW1(LREG)
                 LWLST=LWL(LREG)
                     DO 130 KREG=1,3
                     KW1ST=KW1(KREG)
                     KWLST=KWL(KREG)
                     JW=JW+1
                     WT=0.
                     IF((LW1ST.GT.LWLST).OR.(KW1ST.GT.KWLST)) GO TO 130
                         DO 120 LW=LW1ST,LWLST
                             DO 110 KW=KW1ST,KWLST
110                          WT=WT+WNDW(KW,LW)
120                      CONTINUE
130                  WGT(JW,IW)=WT
140              CONTINUE
150          CONTINUE
160      CONTINUE
         RETURN
         END
```

```
      SUBROUTINE UNIFRM(LVL)
C
C  Purpose: Establish normalized uniform window weights for level LVL
C
C  CALLed from: WGTSET
C
C-----------------------------------------------------------------------
      COMMON /SHARE/ ITEMP(2210)
      DIMENSION WNDW(31,31)
      EQUIVALENCE (WNDW(1,1),ITEMP(1))
      DIMENSION LWNDW(5)
      DATA LWNDW/1,3,7,15,31/
      NWNDW=LWNDW(LVL)
      WPIX=NWNDW*NWNDW
      WT=1./WPIX
      DO 120 L=1,NWNDW
          DO 110 K=1,NWNDW
110       WNDW(K,L)=WT
120   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE GAUSS(LVL)
C
C  Purpose: Establish normalized Gaussian window weights for level LVL
C
C  CALLed from: WGTSET
C
C-----------------------------------------------------------------------
      COMMON /WGTDIT/ KCW,SIG,DIT
      COMMON /SHARE/ ITEMP(2210)
      DIMENSION WNDW(31,31)
      EQUIVALENCE (WNDW(1,1),ITEMP(1))
      DIMENSION LWC(5),LWNDW(5),CORNER(5)
      DATA LWC/1,2,4,8,16/,LWNDW/1,3,7,15,31/,CORNER/0.,1.,3.,7.,15./
      LWCT=LWC(LVL)
      NWNDW=LWNDW(LVL)
      C=CORNER(LVL)
      CSQ=2.*C*C
      FSQ2=-(SIG*SIG)/(2.*CSQ)
      SUM=0.
      DO 120 L=1,NWNDW
      Y=L-LWCT
      YY=Y*Y
          DO 110 K=1,NWNDW
          X=K-LWCT
          XX=X*X
          WT=EXP(FSQ2*(XX+YY))
          SUM=SUM+WT
110       WNDW(K,L)=WT
120   CONTINUE
      DO 220 L=1,NWNDW
          DO 210 K=1,NWNDW
210       WNDW(K,L)=WNDW(K,L)/SUM
220   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SFLOAD
C
C  Purpose: Initialize first target subframe and its surround for each
C           new line-direction subframe row
C             Blank surround along left image edge
C             Load last line of pixels from prior-row subframe
C             Load first line of pixels from following-row subframe
C             Load target subframe
C
C  CALLed from: ADAPT
C
C----------------------------------------------------------------------
      COMMON /SFINPT/ K1ST,KLST,L1ST,LLST,KREC,KBCK,KFWD,NREC
      INTEGER*4 KREC,KBCK,KFWD,NREC
      COMMON /SHARE/ ISFD(65,34),ISFL(65,34)
      BYTE ISFD,ISFL
      LP=LLST+1
      K=K1ST-1
      DO 110 L=1,LP
      ISFD(K,L)=0
110   ISFL(K,L)=0
      DO 120 K=1,KLST
      ISFD(K,1)=0
      ISFD(K,LP)=0
      ISFL(K,1)=0
120   ISFL(K,LP)=0
      KREC=KREC+1                                              ! I*4
      KBCK=KBCK+1                                              ! I*4
      KFWD=KFWD+1                                              ! I*4
      IF(KBCK.LE.1) GO TO 140
      READ (2'KBCK) ((ISFD(K,L),K=K1ST,KLST),L=L1ST,LLST)
      READ (3'KBCK) ((ISFL(K,L),K=K1ST,KLST),L=L1ST,LLST)
      DO 130 K=K1ST,KLST
      ISFD(K,1)=ISFD(K,LLST)
130   ISFL(K,1)=ISFL(K,LLST)
140   IF(KFWD.GT.NREC) GO TO 160
      READ (2'KFWD) ((ISFD(K,L),K=K1ST,KLST),L=L1ST,LLST)
      READ (3'KFWD) ((ISFL(K,L),K=K1ST,KLST),L=L1ST,LLST)
      DO 150 K=K1ST,KLST
      ISFD(K,LP)=ISFD(K,L1ST)
150   ISFL(K,LP)=ISFL(K,L1ST)
160   CONTINUE
      READ (2'KREC) ((ISFD(K,L),K=K1ST,KLST),L=L1ST,LLST)
      READ (3'KREC) ((ISFL(K,L),K=K1ST,KLST),L=L1ST,LLST)
      RETURN
      END
```

```
      SUBROUTINE SFUPDT
C
C  Purpose: Update target subframe and its surround
C                Transfer left surround, prior-line surround, following-
C                  line surround, and target subframe from right surround
C                  area to target area
C                Load last line of pixels from upper right corner
C                  surround subframe
C                Load first line of pixels from lower right corner
C                  surround subframe
C                Load right surround subframe (next target subframe)
C                        Note: "Loads" are replaced by "Blanks" whenever
C                              subframe references would be beyond
C                              image boundaries
C
C  CALLed from: ADAPT
C
C       Geometry of decompressed and resolution image input buffers:
C       -------- -- ------------ --- ---------- ----- ----- --------
C
C                                Pixel Direction  ---->
C
C                               last line of pixels from
C           record KbCK-1   record KBCK     record KBCK+1
C             upper left   upper surround  upper right corner
C               corner    |                |                |
C    L                   *|****************|****************|
C    i                ---+----------------+----------------+
C    n                   *|                |                |
C    e          left     *|                |                |
C             surround   *|                | right surround |
C    D   |               *|     Target     |  (next target  |
C    i   |               *|    Subframe    |   subframe)    |
C    r   |      record   *|                |                |
C    e   V      KREC-1   *| record KREC    | record KREC+1  |
C    c                   *|                |                |
C    t       last column *|                | INPUT REGION,  |
C    i        of pixels  *|                | ALL SUBFRAMES  |
C    o                ---+----------------+----------------+
C    n                   *|****************|****************|
C             lower left   lower surround  lower right corner
C               corner      first line of pixels from
C           record KFWD-1   record KFWD     record KFWD+1
C
C-----------------------------------------------------------------------
      COMMON /SFLOOP/ KSF,NSO,NLS,NPS,KLS,KPS
      COMMUN /SFINPT/ K1ST,KLST,L1ST,LLST,KREC,KBCK,KFWD,NREC
      INTEGER*4 KREC,KBCK,KFWD,NREC
      COMMON /SHARE/ ISFD(65,34),ISFL(65,34)
      BYTE ISFD,ISFL
      LP=LLST+1
      KP=KSF+1
```

```
        DO 120 L=1,LP
        KK=KSF
            DO 110 K=1,KP
            KK=KK+1
            ISFD(K,L)=ISFD(KK,L)
110         ISFL(K,L)=ISFL(KK,L)
120     CONTINUE
        DO 130 L=1,LP
        ISFD(K1ST,L)=0
130     ISFL(K1ST,L)=0
        DO 140 K=K1ST,KLST
        ISFD(K,1)=0
        ISFD(K,LP)=0
        ISFL(K,1)=0
140     ISFL(K,LP)=0
        IF(KPS.EQ.NPS) GO TO 200
        KREC=KREC+1                                          !  I*4
        KBCK=KBCK+1                                          !  I*4
        KFWD=KFWD+1                                          !  I*4
        IF(KBCK.LE.1) GO TO 160
        READ (2'KBCK) ((ISFD(K,L),K=K1ST,KLST),L=L1ST,LLST)
        READ (3'KBCK) ((ISFL(K,L),K=K1ST,KLST),L=L1ST,LLST)
        DO 150 K=K1ST,KLST
        ISFD(K,1)=ISFD(K,LLST)
150     ISFL(K,1)=ISFL(K,LLST)
160     IF(KFWD.GT.NREC) GO TO 180
        READ (2'KFWD) ((ISFD(K,L),K=K1ST,KLST),L=L1ST,LLST)
        READ (3'KFWD) ((ISFL(K,L),K=K1ST,KLST),L=L1ST,LLST)
        DO 170 K=K1ST,KLST
        ISFD(K,LP)=ISFD(K,L1ST)
170     ISFL(K,LP)=ISFL(K,L1ST)
180     CONTINUE
        READ (2'KREC) ((ISFD(K,L),K=K1ST,KLST),L=L1ST,LLST)
        READ (3'KREC) ((ISFL(K,L),K=K1ST,KLST),L=L1ST,LLST)
200     CONTINUE
        RETURN
        END
```

```
      SUBROUTINE SFADPT
C
C  Purpose: Adaptively smooth all MAPSels > 2x2 in target subframe
C               Recursion through all MAPSels in target subframe,
C                 zigzag order
C                   Establish line and pixel indices for sixteen target
C                     and surround elements
C                   Loop on target/surround elements
C                       Retrieve element intensity
C                       Retrieve element level; set activation if
C                         surround element no more than one level below
C                         target MAPSel
C                   Loop on pixels in first (upper left) target MAPSel
C                     quadrant in zigzag order
C                       Determine symmetric target-pixel zigzag addresses
C                         in other three quadrants
C                       Accumulate convolution sums for target pixels
C                         from each quadrant for active elements from
C                         nine associated target/surround regions
C                       Reset (smooth) target pixel from each quadrant
C                         with additive random dither if designated
C                   Return for next MAPSel until subframe is exhausted
C               Output adaptively smoothed subframe
C
C  CALLed from: ADAPT
C
C------------------------------------------------------------------------
      COMMON /WGTDIT/ KCW,SIG,DIT
      COMMON /SFLOOP/ KSF,NSQ,NLS,NPS,KLS,KPS
      COMMON /GAZGIZ/ NZZ(1024)
      COMMON /LPZZ/ LZZ(1024),KZZ(1024)
      COMMON /WGTTBL/ WGT(9,340)
      COMMON /SHARE/ ISFD(65,34),ISFL(65,34)
      BYTE ISFD,ISFL
      COMMON /SFDATA/ ISFA(1024)
      BYTE ISFA
      DIMENSION LVSZ(5),MOSZ(5),MXZZ(5),MSZ(5),MHSZ(5),LVWGT(5),IWT(9,4)
      DIMENSION LSUR(4),KSUR(4),SURI(16),SURL(16),LQZZ(4),SW(4),SWI(4)
      DATA LVSZ/4,16,64,256,1024/
      DATA MOSZ/1,4,16,64,256/
      DATA MXZZ/3,15,63,255,1023/
      DATA MSZ/2,4,8,16,32/
      DATA MHSZ/1,2,4,8,16/
      DATA LVWGT/0,0,4,20,84/
      DATA IWT/1,2,3,5,6,7,9,10,11,  4,3,2,8,7,6,12,11,10,
     +        13,14,15,9,10,11,5,6,7,  16,15,14,12,11,10,8,7,6/
      DATA IR,JR/2*0/
      JZZ=1
100   L=LZZ(JZZ)
      K=KZZ(JZZ)
      LVL=ISFL(K,L)
      IF(LVL.LT.0) LVL=LVL+256
      LV=LVL/32
      IF(LV.LE.1) GO TO 180
```

```
      LSUR(1)=L-1
      LSUR(2)=L
      LSUR(3)=L+MHSZ(LV)
      LSUR(4)=L+MSZ(LV)
      KSUR(1)=K-1
      KSUR(2)=K
      KSUR(3)=K+MHSZ(LV)
      KSUR(4)=K+MSZ(LV)
      JT=0
      DO 120 LT=1,4
      LS=LSUR(LT)
          DO 110 KT=1,4
          KS=KSUR(KT)
          JT=JT+1
          IT=ISFD(KS,LS)
          IF(IT.LT.0) IT=IT+256
          SURI(JT)=IT
          LVT=ISFL(KS,LS)
          IF(LVT.LT.0) LVT=LVT+256
          SURL(JT)=1.
          IF((LVL-LVT).GT.44) SURL(JT)=0.
110       CONTINUE
120   CONTINUE
      JWT=LVWGT(LV)
      NQ=MQSZ(LV)
      DO 170 LOCZZ=1,NQ
      LQZZ(1)=LOCZZ-1
      LQZZ(4)=MXZZ(LV)-LQZZ(1)
      LQZZ(2)=(LQZZ(1).AND."1252).OR.(LQZZ(4).AND."525)
      LQZZ(3)=(LQZZ(1).AND."525).OR.(LQZZ(4).AND."1252)
          DO 130 JQ=1,4
          SW(JQ)=0.
130       SWI(JQ)=0.
          JWT=JWT+1
          DO 150 J=1,9
          WT=WGT(J,JWT)
              DO 140 JQ=1,4
              I=IWT(J,JQ)
              SWT=WT*SURL(I)
              SW(JQ)=SW(JQ)+SWT
140           SWI(JQ)=SWI(JQ)+SWT*SURI(I)
150       CONTINUE
          DO 160 JQ=1,4
          I=IWT(5,JQ)
          AI=SURI(1)
          IF(SW(JQ).GT.0.) AI=SWI(JQ)/SW(JQ)
          IF(DIT.NE.0.) AI=AI+DIT*(RAN(IR,JR)-0.5)
          IT=AI+0.5
          IF(IT.LE.0) IT=0
          IF(IT.GT.255) IT=255
          IF(IT.GT.127) IT=IT-256
          JJZZ=JZZ+LQZZ(JQ)
          IRST=NZZ(JJZZ)
160       ISFA(IRST)=IT
170   CONTINUE
      GO TO 200
```

```
180     IRST=NZZ(JZZ)
        ISFA(IRST)=ISFD(K,L)
        IF(LV.EQ.0) GO TO 200
        DO 190 J=1,3
        JJZZ=JZZ+J
        L=LZZ(JJZZ)
        K=KZZ(JJZZ)
        IRST=NZZ(JJZZ)
190     ISFA(IRST)=ISFD(K,L)
200     INCZZ=1
        IF(LV.NE.0) INCZZ=LVSZ(LV)
        JZZ=JZZ+INCZZ
        IF(JZZ.LE.NSQ) GO TO 100
        WRITE (4) (ISFA(J),J=1,NSQ)
        RETURN
        END
```

SECTION ELEVEN

TRANSMAPS MODULE #5: MAPS DIFFERENCE IMAGE FORMATION

11.1 Program Characteristics

Program Names:	DIFFER or DF	
Subroutines:	USERE FILESE SETUPE SFDIFF ESTATS	
Files:	EPRINT.DAT (listing) IMAGE.DAT or DMAPS.DAT DMAPS.DAT or ADAPT.DAT ERROR.DAT (output)	 (input) (input)
Task Build Options:	MAXBUF = 1024 ACTFIL = 5	
Task Size:	25312	

11.2 Source Listing

The COMMENT-annotated source listing for DIFFER follows:

```
      PROGRAM DIFFER
C
C  +------------------------------------------------------------+
C  |                                                            |
C  |  TransMAPS Module #5:  Difference Image & Statistics Formation  |
C  |                                                            |
C  +------------------------------------------------------------+
C
C                                       Control Data Corporation - 1982
C
C Files: Unit  Name    Content                        From/To  Type
C ------
C   Out    1    EPRINT  Difference statistics report   Printer FORMATTED
C   In     2    IMAGE   Source image, subframes        SUBFRM  FIXED SEQ.
C            or DMAPS   Decompressed image, subframes  DMAPS
C   In     3    DMAPS   Decompressed image, subframes  DMAPS   FIXED SEQ.
C            or ADAPT   Adaptively smoothed, subframes ADAPT
C   Out    4    ERROR   Difference image, subframes    RASTER  FIXED SEQ.
C
C User Interaction:  In Subroutine USERE
C -----------------
C      Image Pair Selection for Differencing
C      Difference Image Control
C
C Program Structure:
C ------------------
C      PROGRAM DIFFER
C          CALL USERE   Specify image pair to difference and output type
C          CALL FILESE   Open files, check integrity, read/write headers
C          CALL SETUPE   Initialize statistical sums and arrays
C          Loop on subframes
C              CALL SFDIFF   Difference images and accumulate statistics
C          CALL ESTATS   Output fidelity measures and image distributions
C
C COMMON Block Communication:
C ---------------------------
C    /DIFF/      User image controls                    Length: 8
C                    DIFFER,USERE,FILESE,SETUPE
C
C      I1TYP(6)          [Byte]  Filename for input image #1
C      I2TYP(6)          [Byte]  Filename for input image #2
C      FAC               [R*4]   Difference image control parameter:
C                                  < 0  form signed and biased difference
C                                  = 0  find difference statistics only
C                                  > 0  form amplified absolute difference
C                                         FAC is amplification factor
C
C    /DIFFOP/    Derived difference controls            Length: 6
C                    SETUPE,SFDIFF
C
C      FACT              [R*4]   Copy of FAC
C      FS                [R*4]   Activation factor for signed difference
C      FA                [R*4]   Activation factor for amplified difference
C
```

```
C     /HEAD1/    Standard MAPS file header, image #1     Length: 32
C                    DIFFER,FILESE
C
C        IFILE1           [I*2]   File type
C        INAME1(8)        [Byte]  User-selected image name
C        NL1              [I*2]   Number of lines in source image
C        NP1              [I*2]   Number of pixels in source image
C        NB1              [I*2]   Number of bits/pixel in source image
C        KSF1             [I*2]   Kind of subframe 8x8 16x16 32x32
C        IGRD1            [I*2]   Subframe grid: square(0)/staggered(1)
C        NS1              [I*4]   Total subframe count
C        MC1              [I*4]   MAPSel count
C        MIXBP1           [I*2]   Packed block(0)/pattern(1) mode (rt-lft)
C        IBV1(5,2)        [I*2]   Optimal pattern biases by level,low/high
C        IPAD1(7)         [I*2]   Space for future extension
C
C     /HEAD2/    Standard MAPS file header, image #2     Length: 32
C                    DIFFER,FILESE
C
C        IFILE2           [I*2]   File type
C        INAME2(8)        [Byte]  User-selected image name
C        NL2              [I*2]   Number of lines in source image
C        NP2              [I*2]   Number of pixels in source image
C        NB2              [I*2]   Number of bits/pixel in source image
C        KSF2             [I*2]   Kind of subframe 8x8 16x16 32x32
C        IGRD2            [I*2]   Subframe grid: square(0)/staggered(1)
C        NS2              [I*4]   Total subframe count
C        MC2              [I*4]   MAPSel count
C        MIXBP2           [I*2]   Packed block(0)/pattern(1) mode (rt-lft)
C        IBV2(5,2)        [I*2]   Optimal pattern biases by level,low/high
C        IPAD2(7)         [I*2]   Space for future extension
C
C     /SFDAT1/   Subframe data, input image #1           Length: 512
C                    SFDIFF
C
C        ISF1(1024)       [Byte]  Image #1 subframe data array
C
C     /SFDAT2/   Subframe data, input image #2           Length: 512
C                    SFDIFF
C
C        ISF2(1024)       [Byte]  Image #2 subframe data array
C
C     /SFDAT3/   Subframe data, difference image         Length: 513
C                    FILESE,SFDIFF
C
C        NSQ              [I*2]   Number of total pixels/subframe
C        ISF3(1024)       [Byte]  Difference image subframe data array
C
C     /STATSE/   Image distributions                     Length: 8576
C                    SETUPE,SFDIFF,ESTATS
C
C        I1VSI2(64,64)    [I*4]   Image #1 vs Image #2 2-D histogram
C        I1H(64)          [I*4]   Image #1 histogram
C        I2H(64)          [I*4]   Image #2 histogram
C        I1M2H(64)        [I*4]   Difference histogram
C                                   (All distributions in steps of
C                                    4 gray-scale level increments)
```

```
C
C     /SUMS/      Fidelity measure accumulations          Length: 20
C                      SETUPE,SFDIFF,ESTATS
C
C          PIX                [R*8]    Total pixel count for subframes
C          SUM1               [R*8]    Sum of image #1 gray-scale values
C          SUMSQ1             [R*8]    Sum of squares of image #1 values
C          DIFF               [R*8]    Sum of difference gray-scale values
C          DIFFSQ             [R*8]    Sum of squares of difference values
C
C-----------------------------------------------------------------------
      COMMON /HEAD1/ IFILE1,INAME1(8),NL1,NP1,NB1,KSF1,IGRD1,NS1,MC1,
     +  MIXBP1,IBV1(5,2),IPAD1(7)
      BYTE INAME1
      INTEGER*4 NS1,MC1
      COMMON /HEAD2/ IFILE2,INAME2(8),NL2,NP2,NB2,KSF2,IGRD2,NS2,MC2,
     +  MIXBP2,IBV2(5,2),IPAD2(7)
      BYTE INAME2
      INTEGER*4 NS2,MC2
      COMMON /DIFF/ I1TYP(6),I2TYP(6),FAC
      BYTE I1TYP,I2TYP
      INTEGER*4 LOOPSF,ISTAR4
      DIMENSION FNAMES(6,4)
      BYTE FNAMES
      DATA FNAMES /1HI,1HM,1HA,1HG,1HE,1H ,1HD,1HM,1HA,1HP,1HS,1H ,
     +  1HL,1HE,1HV,1HE,1HL,1H ,1HA,1HD,1HA,1HP,1HT,1H /
      CALL USERE
      CALL FILESE
      I1=IFILE1
      IF(I1.EQ.0) I1=1
      I2=IFILE2
      IF(I2.EQ.0) I2=1
      TYPE 500,INAME1,(FNAMES(J,I1),J=1,5),INAME2,(FNAMES(J,I2),J=1,5)
500   FORMAT(/,1X,'DIFFERENCING ',8A1,' TYPE ',5A1,', VS ',8A1,
     +  ' TYPE ',5A1)
      CALL SETUPE
      DO 110 LOOPSF=1,NS1                                    ! I*4
110   CALL SFDIFF
      BPP1=NB1
      IF(MC1.EQ.0) GO TO 210
      ISTAR4=(MC1+2)/3                                       ! I*4
      IF(KSF1.EQ.8) ISTAR4=(MC1+3)/4                         ! I*4
      ISTAR4=8*(MC1+ISTAR4)                                  ! I*4
      DEN=NL1
      BPP1=ISTAR4/DEN
      BPP1=BPP1/NP1
210   WRITE (1,600) INAME1,(FNAMES(J,I1),J=1,5),BPP1
600   FORMAT(1H1,5X,'DIFFERENCE STATISTICS:',//,10X,'IMAGE1: ',8A1,
     +  ',  TYPE: ',5A1,',  BITS/PIXEL:',F8.5)
      BPP2=NB2
      IF(MC2.EQ.0) GO TO 220
      ISTAR4=(MC2+2)/3                                       ! I*4
      IF(KSF2.EQ.8) ISTAR4=(MC2+3)/4                         ! I*4
      ISTAR4=8*(MC2+ISTAR4)                                  ! I*4
      DEN=NL2
      BPP2=ISTAR4/DEN
      BPP2=BPP2/NP2
```

```
220     WRITE (1,610) INAME2,(FNAMES(J,I2),J=1,5),BPP2
610     FORMAT(12X,'VS',/,10X,'IMAGE2: ',8A1,',  TYPE: ',5A1,
     +    ',  BITS/PIXEL:',F6.5,/)
        IF(FAC.LT.0.) WRITE (1,620) NS1
620     FORMAT(15X,'SIGNED DIFFERENCE IMAGE PRODUCED:',I7,
     +    ' SUBFRAMES',/)
        IF(FAC.GT.0.) WRITE (1,630) NS1,FAC
630     FORMAT(15X,'AMPLIFIED DIFFERENCE IMAGE PRODUCED:',I7,
     +    ' SUBFRAMES,  AMPLIFICATION',F6.1,/)
        CALL ESTATS
        CLOSE(UNIT=1)
        CLOSE(UNIT=2)
        CLOSE(UNIT=3)
        IF(FAC.NE.0.) CLOSE(UNIT=4)
        END
```

```
        SUBROUTINE USERE
C
C  Purpose: User interaction for difference image and fidelity
C             performance evaluation
C               Subframe image type for input image #1 (IMAGE or DMAPS)
C               Subframe image type for input image #2 (DMAPS or ADAPT)
C               Difference control parameter
C
C  CALLed from: DIFFER
C
C-------------------------------------------------------------------------
        COMMON /DIFF/ I1TYP(6),I2TYP(6),FAC
        BYTE I1TYP,I2TYP
        DIMENSION NAMES(6,3)
        BYTE NAMES
        DATA I1TYP/1HI,1HM,1HA,1HG,1HE,0/                    ! Default
        DATA I2TYP/1HD,1HM,1HA,1HP,1HS,0/                    ! Default
        DATA FAC/10./                                        ! Default
        DATA NAMES/1HI,1HM,1HA,1HG,1HE,0,1HD,1HM,1HA,1HP,1HS,0,
     +    1HA,1HD,1HA,1HP,1HT,0/
        I1=1
        I2=2
        TYPE 500
500     FORMAT(/,1X,32(1H*),/,1X,'*',30X,'*',/,1X,
     +    '* MAPS DIFFERENCE IMAGE MODULE *',
     +    /,1X,'*',30X,'*',/,1X,32(1H*))
100     TYPE 510
510     FORMAT(//,3X,'INPUT IMAGE TYPES:',//,7X,
     +    'I - IMAGE (ORIGINAL SOURCE) IMAGE1 ONLY',/,7X,
     +    'D - DMAPS (MAPS DECOMPRESSED)',/,7X,
     +    'A - ADAPT (ADAPTIVELY SMOOTHED) IMAGE2 ONLY')
        TYPE 520,I1TYP
520     FORMAT(/,5X,'IMAGE1? (I D)  ',A1,'[',4A1,']',A1)
        ACCEPT 1,LIT
1       FORMAT(10A1)
        IF(LIT.EQ.1HI) I1=1
        IF(LIT.EQ.1HD) I1=2
        DO 110 I=1,6
110     I1TYP(I)=NAMES(I,I1)
        IF(I1.NE.2) GO TO 130
        I2=3
        DO 120 I=1,6
120     I2TYP(I)=NAMES(I,I2)
130     TYPE 530,I2TYP
530     FORMAT(5X,'IMAGE2? (D A)  ',A1,'[',4A1,']',A1)
        IF(I1.EQ.2) GO TO 150
        ACCEPT 1,LIT
        IF(LIT.EQ.1HD) I2=2
        IF(LIT.EQ.1HA) I2=3
        DO 140 I=1,6
140     I2TYP(I)=NAMES(I,I2)
```

```
150     TYPE 540,FAC
540     FORMAT(/,3X,'DIFFERENCE IMAGE TYPE:',//,7X,
     +    'AMPLIFICATION FACTOR',/,9X,
     +    '<0  SIGNED AND BIASED DIFFERENCE IMAGE',/,9X,
     +    '=0  NO DIFFERENCE IMAGE, STATISTICS ONLY',/,9X,
     +    '>0  AMPLIFIED DIFFERENCE IMAGE',/,15X,
     +         '(VALUE IS AMPLIFICATION)',//,5X,
     +    'FACTOR?',F5.0,5X,'(/ = NO CHNG)')
        ACCEPT *,FAC
        TYPE 550
550     FORMAT(//,3X,'USER SPECIFICATION COMPLETE:',/,3X,28(1H*),//,5X,
     +    'REVIEW? (Y OR N)  N')
        ACCEPT 1,LIT
        IF(LIT.EQ.1HY) GO TO 100
        RETURN
        END
```

```
      SUBROUTINE FILESE
C
C  Purpose: Open files, check integrity, and read/write file headers
C
C  CALLed from: DIFFER
C
C-----------------------------------------------------------------------
      COMMON /HEAD1/ IFILE1,INAME1(8),NL1,NP1,NB1,KSF1,IGRD1,NS1,MC1,
     +  MIXBP1,IBV1(5,2),IPAD1(7)
      BYTE INAME1
      INTEGER*4 NS1,MC1
          DIMENSION JHEAD1(32)
          EQUIVALENCE (JHEAD1(1),IFILE1)
      COMMON /HEAD2/ IFILE2,INAME2(8),NL2,NP2,NB2,KSF2,IGRD2,NS2,MC2,
     +  MIXBP2,IBV2(5,2),IPAD2(7)
      BYTE INAME2
      INTEGER*4 NS2,MC2
          DIMENSION JHEAD2(32)
          EQUIVALENCE (JHEAD2(1),IFILE2)
      COMMON /DIFF/ I1TYP(6),I2TYP(6),FAC
      BYTE I1TYP,I2TYP
      COMMON /SFDAT3/ NSO,ISF3(1024)
      BYTE ISF3
          DIMENSION JSF(512)
          EQUIVALENCE (JSF(1),ISF3(1))
      OPEN(UNIT=2,TYPE='OLD',NAME=I1TYP,FORM='UNFORMATTED',
     +  RECORDTYPE='FIXED')
      READ (2) JHEAD1
      I1=IFILE1
      IF((I1.EQ.0).OR.((I1.GE.2).AND.(I1.LE.4))) GO TO 110
              IT=1
              TYPE 500,IT,I1
500           FORMAT(/,1X,'*** FILE',I2,' TYPE',I2,' INVALID INPUT')
              STOP 1
110   OPEN(UNIT=3,TYPE='OLD',NAME=I2TYP,FORM='UNFORMATTED',
     +  RECORDTYPE='FIXED')
      READ (3) JHEAD2
      I2=IFILE2
      IF((I2.EQ.0).OR.((I2.GE.2).AND.(I2.LE.4))) GO TO 120
              IT=2
              TYPE 500,IT,I2
              STOP 2
120   IF(KSF1.EQ.KSF2) GO TO 130
              TYPE 510,KSF1,KSF2
510           FORMAT(/,1X,'*** SUBFRAME SIZES DISAGREE:',I3,' VS',I3)
              STOP
130   IF(NS1.EQ.NS2) GO TO 200
              TYPE 520,NS1,NS2
520           FORMAT(/,1X,'*** SUBFRAME COUNTS DISAGREE:',I7,' VS',I7)
              STOP
```

```
200     NSQ=KSF1*KSF1
        IF(FAC.EQ.0.) GO TO 250
        LSQ=NSQ/4
        OPEN(UNIT=4,TYPE='NEW',NAME='ERROR',FORM='UNFORMATTED',
     +    RECORDTYPE='FIXED',RECORDSIZE=LSQ)
        DO 210 J=1,32
210     JSF(J)=JHEAD1(J)
        IFILE3=5
        IF(I2TYP(1).EQ.1HA) IFILE3=6
        IF(I1TYP(1).EQ.1HD) IFILE3=7
        JSF(1)=IFILE3
        WRITE (4) (ISF3(J),J=1,NSQ)
        DO 220 J=1,32
220     JSF(J)=0
250     OPEN(UNIT=1,TYPE='NEW',NAME='EPRINT')
        RETURN
        END
```

```
      SUBROUTINE SETUPE
C
C  Purpose: Initialize statistical sums and accumulation arrays
C
C  CALLed from: DIFFER
C
C-----------------------------------------------------------------------
      COMMON /DIFF/ I1TYP(6),I2TYP(6),FAC
      BYTE I1TYP,I2TYP
      COMMON /DIFFOP/ FACT,FS,FA
      COMMON /SUMS/ PIX,SUM1,SUMSQ1,DIFF,DIFFSQ
      REAL*8 PIX,SUM1,SUMSQ1,DIFF,DIFFSQ
      COMMON /STATSE/ I1VSI2(64,64),I1H(64),I2H(64),I1M2H(64)
      INTEGER*4 I1VSI2,I1H,I2H,I1M2H
      FACT=FAC
      FS=1.
      FA=0.
      IF(FACT.LE.0.) GO TO 110
      FS=0.
      FA=FACT
110   CONTINUE
      PIX=0.D0
      SUM1=0.D0
      SUMSQ1=0.D0
      DIFF=0.D0
      DIFFSQ=0.D0
      DO 130 J=1,64
          DO 120 I=1,64
120       I1VSI2(I,J)=0
      I1H(J)=0
      I2H(J)=0
130   I1M2H(J)=0
      RETURN
      END
```

```
      SUBROUTINE SFDIFF
C
C  Purpose: Accumulate difference statistics and generate difference
C           image subframe
C
C  CALLed from: DIFFER
C
C------------------------------------------------------------------------
      COMMON /DIFFOP/ FACT,FS,FA
      COMMON /SUMS/ PIX,SUM1,SUMSQ1,DIFF,DIFFSQ
      REAL*8 PIX,SUM1,SUMSQ1,DIFF,DIFFSQ
      COMMON /STATSE/ I1VSI2(64,64),I1H(64),I2H(64),I1M2H(64)
      INTEGER*4 I1VSI2,I1H,I2H,I1M2H
      COMMON /SFDAT1/ ISF1(1024)
      BYTE ISF1
      COMMON /SFDAT2/ ISF2(1024)
      BYTE ISF2
      COMMON /SFDAT3/ NSQ,ISF3(1024)
      BYTE ISF3
      READ (2) (ISF1(J),J=1,NSQ)
      READ (3) (ISF2(J),J=1,NSQ)
      DO 110 I=1,NSQ
      I1=ISF1(I)
      IF(I1.LT.0) I1=I1+256
      I2=ISF2(I)
      IF(I2.LT.0) I2=I2+256
      I1M2=I1-I2
      PIX=PIX+1.D0
      GS1=I1
      SUM1=SUM1+GS1
      SUMSQ1=SUMSQ1+GS1*GS1
      GS1M2=I1M2
      DIFF=DIFF+GS1M2
      DIFFSQ=DIFFSQ+GS1M2*GS1M2
      J1=I1/4+1
      J2=I2/4+1
      J1M2=I1M2/4+33
      IF(J1M2.LT.1) J1M2=1
      IF(J1M2.GT.64) J1M2=64
      I1VSI2(J1,J2)=I1VSI2(J1,J2)+1                        !  I*4
      I1H(J1)=I1H(J1)+1                                    !  I*4
      I2H(J2)=I2H(J2)+1                                    !  I*4
      I1M2H(J1M2)=I1M2H(J1M2)+1                            !  I*4
      IPOS=I1M2
      IF(IPOS.LT.0) IPOS=-IPOS
      IDIFF=FS*(I1M2+127)+FA*IPOS
      IF(IDIFF.LE.0) IDIFF=0
      IF(IDIFF.GT.255) IDIFF=255
      IF(IDIFF.GT.127) IDIFF=IDIFF-256
      ISF3(I)=IDIFF
110   CONTINUE
      IF(FACT.NE.0.) WRITE (4) (ISF3(J),J=1,NSQ)
      RETURN
      END
```

```
      SUBROUTINE ESTATS
C
C  Purpose. Output fidelity measures and image distributions
C
C  CALLed from: DIFFER
C
C-----------------------------------------------------------------
      COMMON /SUMS/ PIX,SUM1,SUMSQ1,DIFF,DIFFSQ
      REAL*8 PIX,SUM1,SUMSQ1,DIFF,DIFFSQ
      COMMON /STATSE/ I1VSI2(64,64),I1H(64),I2H(64),I1M2H(64)
      INTEGER*4 I1VSI2,I1H,I2H,I1M2H
      DIMENSION LBLL(64),LINE(64),LSYMB(11)
      INTEGER*4 NRMPIX,ISTAR4
      REAL*8 SSQREL
      DATA LBLL/29*1H ,1HI,1HM,1HA,1HG,1HE,1H ,1H2,28*1H /
      DATA LSYMB/1H.,1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9,1H*/
      AVDIFF=DIFF/PIX
      RATIO=DIFFSQ/PIX
      RMS=SQRT(RATIO)
      ERR=0.
      IF(SUMSQ1.GT.0.D0) ERR=100.*(DIFFSQ/SUMSQ1)
      REL=0.
      SSQREL=SUMSQ1-SUM1*SUM1/PIX
      IF(SSQREL.GT.0.D0) REL=100.*(DIFFSQ/SSQREL)
      WRITE (1,640) AVDIFF,RMS,ERR,REL
640   FORMAT(10X,'DIFFERENCES, IMAGE1-IMAGE2: MEAN',F8.3,',  RMS',
     +  F8.3,',  MSE',F9.5,'%,  RMSE',F9.5,'%',/)
      NRMPIX=PIX/640.D0                                     !  I*4
      WRITE (1,700) NRMPIX
700   FORMAT(20X,'IMAGE 1    (UNIT =',I6,' PIXELS)',40X,
     +  'DISTRIBUTIONS',/,70X,'GRAY SCALE  IMAGE 1  IMAGE 2',4X,
     +  'DIFFERENCE IMAGE1-IMAGE2',/,3X,'+',64(1H-),'+')
      DO 150 J=1,64
          DO 110 I=1,64
110       LINE(I)=1H
          LINE(J)=1H+
          DO 120 I=1,64
          ISTAR4=I1VSI2(I,J)
          IF(ISTAR4.EQ.0) GO TO 120
          II=ISTAR4/NRMPIX+1                                !  I*4
          IF(II.GT.10) II=11
          LINE(I)=LSYMB(II)
120       CONTINUE
      LL=4*(J-1)
      LU=LL+3
      LDL=4*(J-33)
      LDU=LDL+3
      WRITE (1,710) LBLL(J),LINE,LL,LU,I1H(J),I2H(J),LDL,LDU,I1M2H(J)
710   FORMAT(1X,A1,' |',64A1,'|',I4,' TO',I4,2I9,I6,' TO',I5,I11)
150   CONTINUE
      WRITE (1,720)
720   FORMAT(3X,'+',64(1H-),'+')
      RETURN
      END
```

SECTION TWELVE

TRANSMAPS MODULE #6: SUBFRAME TO RASTER CONVERSION

12.1 Program Characteristics

Program Names: RASTER or RS

Subroutines:	USERR	
	FILESR	
	SETUPR	
	UNSQR	
	UNSTGR	
Files:	IMAGE.DAT	(input)
	or DMAPS.DAT	
	or LEVEL.DAT	
	or ADAPT.DAT	
	or ERROR.DAT	
	IRAST.DAT	(output)
	or DRAST.DAT	
	or LRAST.DAT	
	or ARAST.DAT	
	or ERAST.DAT	
Task Build Options:	MAXBUF = 1024	
Task Size:	29664	

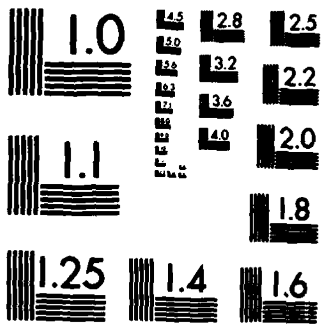
12.2 Source Listing

The COMMENT-annotated source listing for RASTER follows:

AD-A120 982   TRANSPORTABLE MAPS SOFTWARE VOLUME I(U) CONTROL DATA
CORP MINNEAPOLIS MN INFORMATION SCIENCES DIV
A E LABONTE JUL 82 9199500 RADC-TR-82-194

UNCLASSIFIED   F30602-80-C-0326   F/G 20/6   NL

END
FILMED
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

```
      PROGRAM RASTER
C
C  +------------------------------------------------+
C  !                                                !
C  !  TransMAPS Modul: #6:  Subframe to Raster Conversion  !
C  !                                                !
C  +------------------------------------------------+
C
C                                      Control Data Corporation - 1982
C
C Files: Unit  Name     Content                          From/To  Type
C ------
C   In     2     IMAGE   Source image, subframes          SUBFRM  DIRECT
C             or DMAPS   Decompressed image, subframes    DMAPS
C             or LEVEL   Resolution image, subframes      DMAPS
C             or ADAPT   Adaptively smoothed, subframes   ADAPT
C             or ERROR   Difference image, subframes      DIFFER
C   Out    3     IRAST   Source image, raster             User    SEGMENTED
C             or DRAST   Decompressed image, raster
C             or LRAST   Resolution image, raster
C             or ARAST   Adaptively smoothed, raster
C             or ERAST   Difference image, raster
C
C User Interaction:  In Subroutine USERR
C -----------------
C       Image Type Selection for Conversion
C
C Program Structure:
C ------------------
C       PROGRAM RASTER
C           CALL USERR   Specify image to be converted
C           CALL FILESR   Open files, read header, check type integrity
C           CALL SETUPR   Establish partition controls, check image
C                           size integrity
C           IF square grid:
C               CALL UNSQR   Convert 8x8 16x16 32x32 subframes to raster
C                   Loop on rows of subframes in line direction
C                       Loop on 8-line swathes within rows
C                           Loop on subframes in pixel direction
C                               Input subframe
C                               Extract line segments for swath
C                           Output completed lines (burst of 8)
C        or IF staggered grid:
C               CALL UNSTGR   Convert 8x8 staggered subframes to raster
C                   Preload buffer with all subframes with first line
C                   Loop on lines
C                       Output current line
C                       Replace all subframes needed to finish next line
C
C COMMON Block Communication:
C ---------------------------
C    .Blank.     Raster image data                      Length: 16000
C                   RASTER,UNSQR,UNSTGR
C
C       IBUF(4000,8)     (Byte)  Block or recirculating 8-line buffer
C
```

```
C     /DSQR/      Square grid partition controls          Length: 3
C                     SETUPR,UNSQR
C
C        NPS              [I*2]   Number of pixel-direction subframes
C        NLS              [I*2]   Number of line-direction subframes
C        NSWTH            [I*2]   Number of swathes/subframe (1 2 4)
C
C     /DSTGR/     Staggered grid partition controls       Length: 17
C                     SETUPR,UNSTGR
C
C        NPST             [I*2]   Number of pixel-direction subframes total
C        NPSL(8)          [I*2]   Number of staggered subframes in pixel
C                                   direction for each startline mod 8
C        IPSL(8)          [I*2]   Initial pixel of first staggered
C                                   subframe for each startline mod 8
C
C     /HEADER/    Standard MAPS file header               Length: 32
C                     RASTER,FILESR,SETUPR
C
C        IFILE            [I*2]   File type
C        INAME(8)         [Byte]  User-selected image name
C        NL               [I*2]   Number of lines in source image
C        NP               [I*2]   Number of pixels in source image
C        NB               [I*2]   Number of bits/pixel in source image
C        KSF              [I*2]   Kind of subframe 8x8 16x16 32x32
C        IGRD             [I*2]   Subframe grid: square(0)/staggered(1)
C        NS               [I*4]   Total subframe count
C        MC               [I*4]   MAPSel count
C        MIXBP            [I*2]   Packed block(0)/pattern(1) mode (rt-lft)
C        IBV(5,2)         [I*2]   Optimal pattern biases by level,low/high
C        IPAD(7)          [I*2]   Space for future extension
C
C     /OUTNM/     Output file identification              Length: 3
C                     RASTER,USERR,FILESR
C
C        NAMER(6)         [Byte]  Output file name
C
C     /SFDATA/    Subframe data                           Length: 512
C                     UNSQR,UNSTGR
C
C        ISF(1024)        [Byte]  Subframe data input array
C
C     /SFTEMP/    Subframe control parameters             Length: 4
C                     SETUPR,UNSQR,UNSTGR
C
C        NLT              [I*2]   Number of lines in image
C        NPT              [I*2]   Number of pixels in image
C        KSFT             [I*2]   Subframe size: edge
C        KSQT             [I*2]   Subframe size: pixel count
C
C     /SUBFNM/    Input file identification               Length: 3
C                     USERR,FILESR
C
C        NAMEF(6)         [Byte]  Input file name
C
C-------------------------------------------------------------------------
```

```
      COMMON IBUF(4000,8)
      BYTE IBUF
      COMMON /HEADER/ IFILE,INAME(8),NL,NP,NB,KBF,IGRD,NS,NC,NIXBP,
     +  IBV(5,2),IPAD(7)
      BYTE INAME
      INTEGER*4 NS,NC
      COMMON /OUTNM/ NAMER(6)
      BYTE NAMER
      DIMENSION NAMEIN(9,8)
      BYTE NAMEIN
      DATA NAMEIN/1HI,1HM,1HA,1HG,1HE,1H ,1H ,1H ,1H ,
     +           1H ,1H ,1H ,1H ,1H ,1H ,1H ,1H ,1H ,
     +           1HD,1HM,1HA,1HP,1HS,1H ,1H ,1H ,1H ,
     +           1HL,1HE,1HV,1HE,1HL,1H ,1H ,1H ,1H ,
     +           1HA,1HD,1HA,1HP,1HT,1H ,1H ,1H ,1H ,
     +           1HE,1HR,1HR,1HO,1HR,1H ,1HD,1H-,1HI,
     +           1HE,1HR,1HR,1HO,1HR,1H ,1HA,1H-,1HI,
     +           1HE,1HR,1HR,1HO,1HR,1H ,1HA,1H-,1HD/
      CALL USERR
      CALL FILESR
      CALL SETUPR
      JFILE=IFILE+1
      TYPE 500,INAME,(NAMEIN(J,JFILE),J=1,9),NL,NP,NAMER
500   FORMAT(/,1X,'CONVERTING IMAGE ',8A1,', FILE TYPE: ',9A1,//,
     +  3X,'TO',I5,' LINE BY',I5,' PIXEL RASTER, FILE TYPE: ',6A1)
      IF(IGRD.NE.0) GO TO 110
      CALL UNSOR
      GO TO 120
110   CALL UNSTGR
120   CONTINUE
      CLOSE(UNIT=2)
      CLOSE(UNIT=3)
      END
```

```
      SUBROUTINE USERR
C
C  Purpose: User interaction for subframe to raster conversion
C             image type to be converted
C
C  CALLed from: RASTER
C
C-----------------------------------------------------------------------
      COMMON /SUBFRM/ NAMEF(6)
      BYTE NAMEF
      COMMON /OUTNM/ NAMEP(6)
      BYTE NAMEP
      DIMENSION NAMES(6,5)
      BYTE NAMES
      DATA NAMEF/1HD,1HM,1HA,1HP,1HS,0/                         ! Default
      DATA NAMEP/1HD,1HR,1HA,1HS,1HT,0/                         ! Default
      DATA NAMES/1HI,1HM,1HA,1HG,1HE,0,1HD,1HM,1HA,1HP,1HS,0,
     *  1HL,1HE,1HV,1HE,1HL,0,1HA,1HD,1HA,1HP,1HT,0,
     *  1HE,1HR,1HR,1HO,1HR,0/
      IOUT=2                                                    ! Default
      TYPE 500
500   FORMAT(/,1X,45(1H*),/,1X,'*',43X,'*',/,1X,
     *  '* MAPS SUBFRAME TO RASTER CONVERSION MODULE *',
     *  /,1X,'*',43X,'*',/,1X,45(1H*))
100   TYPE 510,NAMEF
510   FORMAT(//,3X,'MAPS PRODUCT IMAGE TYPE:',//,7X,
     *  'I - IMAGE (ORIGINAL SOURCE)',/,7X,
     *  'D - DMAPS (MAPS DECOMPRESSED)',/,7X,
     *  'L - LEVEL (MAPS RESOLUTION CODES)',/,7X,
     *  'A - ADAPT (ADAPTIVELY SMOOTHED)',/,7X,
     *  'E - ERROR (DIFFERENCE)',//,5X,
     *  'TYPE? (I D L A E)  ',A1,'[',4A1,']',A1)
      ACCEPT 1,LIT
1     FORMAT(10A1)
      IF(LIT.EQ.1HI) IOUT=1
      IF(LIT.EQ.1HD) IOUT=2
      IF(LIT.EQ.1HL) IOUT=3
      IF(LIT.EQ.1HA) IOUT=4
      IF(LIT.EQ.1HE) IOUT=5
      DO 110 I=1,6
110   NAMEF(I)=NAMES(I,IOUT)
      NAMEP(1)=NAMEF(1)
      TYPE 520
520   FORMAT(//,3X,'USER SPECIFICATION COMPLETE:',/,3X,28(1H*),//,5X,
     *  'REVIEW? (Y OR N)  N')
      ACCEPT 1,LIT
      IF(LIT.EQ.1HY) GO TO 100
      RETURN
      END
```

```
      SUBROUTINE FILESR
C
C  Purpose: Open files, read header, check type integrity
C
C  CALLed from: RASTER
C
C-----------------------------------------------------------------
      COMMON /HEADER/ IFILE,INAME(8),NL,NP,NB,NBF,IORD,NS,NC,NIIDP,
     *  IDV(5,2),IPAD(7)
      BYTE INAME
      INTEGER*4 NS,NC
              DIMENSION JHEAD(32)
              EQUIVALENCE (JHEAD(1),IFILE)
      COMMON /SUBFNM/ NAMEF(6)
      BYTE NAMEF
      COMMON /OUTNM/ NAMER(6)
      BYTE NAMER
      OPEN(UNIT=2,TYPE='OLD',NAME=NAMEF,FORM='UNFORMATTED',
     *  RECORDTYPE='FIXED',ACCESS='DIRECT')
      READ (2'1) JHEAD
      IF((IFILE.LT.0).OR.(IFILE.EQ.1).OR.(IFILE.GT.7)) GO TO 110
      OPEN(UNIT=3,TYPE='NEW',NAME=NAMER,FORM='UNFORMATTED')
      GO TO 120
110           TYPE 900,IFILE,INAME
900           FORMAT(1X,'*** INVALID FILE TYPE: IFILE =',I6,
     *          ',  IMAGE: ',8A1)
              STOP
120   RETURN
      END
```

```
      SUBROUTINE SETUPR
C
C  Purpose: Establish conversion controls
C               Establish subframe partition parameters
C               Check image size integrity
C
C  CALLed from: RASTER
C
C-----------------------------------------------------------------------
      COMMON /HEADER/ IFILE,INAME(8),NL,NP,NB,NSF,IGRD,NS,NC,NILDP,
     *   IBV(5,2),IPAD(7)
      BYTE INAME
      INTEGER*4 NS,NC
               INTEGER*4 NST
      COMMON /DSGR/ NPS,NLS,NSWTH
      COMMON /DSTGR/ NPST,NPSL(8),IPSL(8)
               DIMENSION JPSL(8),JLSL(8)
      COMMON /SFTEMP/ NLT,NPT,NSFT,NSST
      INTEGER*4 ISTAR4
      DATA IPSL/1,25,49,9,33,57,17,41/
      DATA JPSL/7,4,1,6,3,0,5,2/,JLSL/7,14,13,12,11,10,9,8/
      NPS=(NP-1)/NSF+1
      IF(IGRD.NE.0) GO TO 110
C
C     Square Grid Partition
C
      NLS=(NL-1)/NSF+1
      ISTAR4=NPS
      NST=NLS*ISTAR4                                          I  I*4
      IF(NS.NE.NST) GO TO 150
      NSWTH=1
      IF(NSF.EQ.16) NSWTH=2
      IF(NSF.EQ.32) NSWTH=4
      GO TO 130
C
C     Staggered Grid Partition
C
110   NPST=NPS
      NST=0
      DO 120 J=1,8
      NPSL(J)=(NPS+JPSL(J))/8
      ISTAR4=NPSL(J)
120   NST=NST+((NL+JLSL(J))/8)*ISTAR4                        I  I*4
      IF(NS.NE.NST) GO TO 150
130   NLT=NL
      NPT=NP
      NSFT=NSF
      NSST=NSF*NSF
      GO TO 200
150        TYPE 900,NL,NP,NSF,NSF,IGRD,NS
900        FORMAT(1X,'***',I5,' LINES,',I5,' PIXELS,',I3,' X',I3,
     *        ' SUBFRAMES, GRID',I2,/,4X,'INCONSISTENT WITH',I7,
     *        ' SUBFRAME COUNT')
           STOP
200   RETURN
      END
```

```
      SUBROUTINE UNSQR
C
C  Purpose: Convert square-grid subframes to raster image
C                 Loop on rows of subframes in line direction
C                     Loop on 8-line swathes within rows
C                         Loop on subframes in pixel direction
C                             Input subframe
C                             Extract line segments for swath
C                         Output completed lines (bursts of 8)
C
C  Called from: RASTER
C
C-----------------------------------------------------------------------
      COMMON IBUF(4000,8)
      BYTE IBUF
      COMMON /SQSM/ NPS,NLS,NSWTH
      COMMON /SFTEMP/ NLT,NPT,NSFT,NSQT
      COMMON /SFDATA/ ISF(1024)
      BYTE ISF
      INTEGER*4 NREC,JREC
      L=0
      NREC=1
      DO 300 JLS=1,NLS
          DO 220 JSWTH=1,NSWTH
              JREC=NREC
              J1P=1
              JLP=NSFT
              ISOT=8*NSFT*(JSWTH-1)
              DO 130 JPS=1,NPS
              JREC=JREC+1                                    !  I*4
                  READ (2'JREC) (ISF(JSQT),JSQT=1,NSQT)
                  JSQT=ISOT
                  DO 120 JL1=1,8
                      DO 110 JP1=J1P,JLP
                      JSQT=JSQT+1
110                   IBUF(JP1,JL1)=ISF(JSQT)
120               CONTINUE
              J1P=J1P+NSFT
130           JLP=JLP+NSFT
              DO 210 JL1=1,8
                  WRITE (3) (IBUF(JP1,JL1),JP1=1,NPT)
              L=L+1
              IF(L.GE.NLT) GO TO 400
210           CONTINUE
220       CONTINUE
300   NREC=NREC+NPS                                          !  I*4
400   RETURN
      END
```

```
      SUBROUTINE UNSTGR
C
C  Purpose: Convert staggered-grid subframes to raster image
C              Preload recirculating buffer with all subframes
C                containing segments from first line
C              Loop on lines
C                  Output current line
C                  Replace all subframes needed to complete next line
C
C  Called from: RASTER
C
C-----------------------------------------------------------------------
      COMMON IBUF(4096,8)
      BYTE IBUF
      COMMON /RSTGR/ NPST,NPSL(8),IPSL(8)
      COMMON /SFTEMP/ NLT,NPT,NSFT,NSUT
      COMMON /SFDATA/ ISF(1024)
      BYTE ISF
      INTEGER*4 JREC
      JREC=1
      DO 150 LL=1,8
      [illegible]
      [illegible]
      [illegible]
      IF(JPSL.EQ.0) GO TO 150
      [illegible]
      [illegible]
         DO 130 [illegible]
         JREC=JREC+1                                          1   [illegible]
            READ (2'JREC) (ISF(JSUT),[illegible])
         [illegible]
         [illegible]
            DO 120 [illegible]
               DO 110 [illegible]
               [illegible]
110            IBUF([illegible])=ISF([illegible])
            [illegible]
            IF([illegible].GT.8) [illegible]
120         CONTINUE
         [illegible]
130      [illegible]
150   CONTINUE
      DO 250 [illegible]
      [illegible]
      [illegible]
      IF([illegible].EQ.0) [illegible]
         WRITE (3) (IBUF([illegible]),[illegible])
```

```
        IF(L.GE.NLT) GO TO 300
        JPGL=NPGL(LSI)
        IF(JPGL.EQ.0) GO TO 250
        J1P=IPGL(LSI)
        JLP=J1P+7
            DO 230 JPG=1,JPGL
            JREC=JREC+1                                       I   IO4
                READ (2'JREC) (IBF(JGOT),JGOT=1,NGOT)
            JLIN=LSI
            JGOT=0
                DO 220 JLT=1,8
                    DO 210 JPIX=J1P,JLP
                    JGOT=JGOT+1
210                 IBUF(JPIX,JLIN)=IBF(JGOT)
                JLIN=JLIN+1
                IF(JLIN.GT.8) JLIN=1
220             CONTINUE
            J1P=J1P+64
230         JLP=JLP+64
250     CONTINUE
300     RETURN
        END
```

SECTION THIRTEEN

TRANSMAPS MODULE #7: IMAGE ASSEMBLY AND ANNOTATION

13.1 Program Characteristics

Program Names: ANNOTE or AI

Subroutines: USER
UMSG
OVRLAP
SINDEX
SYMBIN
FILSET
TXTSET
INDEX4
LINES4
CLMNS4
RSMPL1
RSMPL2
RSMPL3
RSMPL4

Files: SYMBOL.DAT (input)
User1.DAT (input raster image)
User2.DAT (input raster image)
ANNIMG.DAT (output)
or ANPRNT.DAT (listing)

Task Build Options: none

Task Size: 31000

13.2 Source Listing

The COMMENT-annotated source listing for ANNOTE follows:

```
      PROGRAM ANNOTE
C
C  +-----------------------------------------------------+
C  !                                                     !
C  !  TransMAPS Module #7:  Image Assembly and Annotation !
C  !                                                     !
C  +-----------------------------------------------------+
C
C                                   Control Data Corporation - 1982
C
C Files: Unit  Name    Content                        From/To  Type
C ------
C   In    1    SYMBOL  Symbol Table                   System   SEGMENTED
C   In    2    User    Embedded Image #1              User     SEGMENTED
C   In    3    User    Embedded Image #2              User     SEGMENTED
C   Out   4    ANIMG   Output Raster Image            User     SEGMENTED
C           or APRINT  Line Printer Pseudo-Image      Printer  FORMATTED
C
C User Interaction:  in Subroutines USER & UMSG
C ------------------
C       Output Image Specifications
C       Embedded (Input) Image Count (0-2) and Specifications
C       Embedded Annotation Message Count (0-20) and Specifications
C
C Program Structure:
C ------------------
C       PROGRAM ANNOTE
C          CALL USER    Specify Output Image and Embedded Images
C             CALL UMSG(N)    Specify Annotation - Message N
C             CALL UVRLAP     Report message conflicts (overwrites)
C                CALL UMSG(N1)    Edit overwritten message if desired
C                CALL UMSG(N2)    Edit overwriting message if desired
C          CALL SIMBOL   Convert messages from ASCII to Symbol Indices
C          CALL SYMBIN   Input Symbol Map Tables
C          CALL FILSET   Open and position image files
C          Loop on Output Lines (L):
C             Loop on Embedded Images:
C                Transfer Embedded Image line to Output Image line
C             CALL TITSET(L)   Get Annotation for current line
C                Loop on Messages:
C                   IF "T" or "B" Orientation:
C                      CALL INDEX4   Determine Symbol Map lines
C                      CALL LINES4   Extract Symbol Map lines
C                      CALL RSMPL1   Resample to 1x
C                   or CALL RSMPL2   Resample to 2x
C                   or CALL RSMPL3   Resample to 3x
C                   or CALL RSMPL4   Resample to 4x
C                or IF "L" or "R" Orientation:
C                      CALL INDEX4   Determine Symbol Map columns
C                      CALL COLMN4   Extract Symbol Map columns
C                      CALL RSMPL1   Resample to 1x
C                   or CALL RSMPL2   Resample to 2x
C                   or CALL RSMPL3   Resample to 3x
C                   or CALL RSMPL4   Resample to 4x
C                   Transfer annotation segment to Output Image line
C             Write Output Image line
```

```
C
C COMMON Block Communication.
C --------------------------
C    /EMBIMG/   Embedded image specification           Length: 31
C               ANNOTE,USER,OVRLAP,FILSET
C
C       NIMG           (I*2)   Number of Embedded Images (0-2)
C       FILNAM(10,2)   (Byte)  User input file names
C       JL(2)          (I*2)   Input lines to skip
C       JP(2)          (I*2)   Input pixels to skip
C       NL(2)          (I*2)   Input lines to transfer
C       NP(2)          (I*2)   Input pixels to transfer
C       INP(2)         (I*2)   Total input pixels (skip + transfer)
C       IL(2)          (I*2)   Initial (starting) line on Output Image
C       LL(2)          (I*2)   Last line on Output Image
C       IP(2)          (I*2)   Initial (starting) pixel on Output Image
C       LP(2)          (I*2)   Last pixel on Output Image
C       IC(2)          (I*2)   Input image complement select (non-0)
C
C    /EMGMSG/   Annotation specification               Length: 641
C               USER,UMSG,OVRLAP,SINDEX,TXTSET
C
C       NMSG           (I*2)   Number of Messages (0-20)
C       MSGO(20)       (I*2)   Message orientation (TBLR)
C       MSGN(20)       (I*2)   Message character count (50 maximum)
C       MSGM(20)       (I*2)   Message symbol size (1x 2x 3x 4x)
C       NIS(20)        (I*2)   Symbol size in pixels
C       MSGL(20)       (I*2)   Message center line on Output Image
C       MSGP(20)       (I*2)   Message center pixel on Output Image
C       MSGC(20)       (I*2)   Message complement select (non-0)
C       TEXT(50,20)    (Byte)  Message text
C
C    /IMGMSG/   Embedded Image/Message Assembly        Length: 2000
C               ANNOTE,TXTSET
C
C       LTMP(4000)     (Byte)  Temporary staging array
C
C    /LCLMSG/   Local message control parameters       Length: 44
C               TXTSET,INDEX4,LINES4,CLMNS4,
C               RSMPL1,RSMPL2,RSMPL3,RSMPL4
C
C       MZ             (I*2)   Message symbol size
C       ML             (I*2)   Line within message frame
C       ML4            (I*2)   Number of lines/columns from Symbol Map
C       MI4(2)         (I*2)   Indices of lines/columns from Symbol Map
C       NT             (I*2)   Current symbol index
C       LAT(3,2)       (I*2)   Extracted lines/columns from Symbol Map
C       CHAR(64)       (Byte)  Annotation pixels for this symbol & line
C
C    /MBOUND/   Message line and pixel ranges          Length: 80
C               USER,UMSG,OVRLAP,TXTSET
C
C       MIL(20)        (I*2)   Message initial line on Output Image
C       MLL(20)        (I*2)   Message last line on Output Image
C       MIP(20)        (I*2)   Message initial pixel on Output Image
C       MLP(20)        (I*2)   Message last pixel on Output Image
```

```
C
C     /MODEU4/    Output image mode                         Length: 1
C                     ANNOTE,USER,FILSET
C
C        IPRNT               (I*2)    Raster image(0)/Printer pseudo-image(1)
C
C     /OUTIMG/    Output Image specification                Length: 3
C                     ANNOTE,USEP,UNSG,OVRLAP
C
C        NL                  (I*2)    Number of lines in Output Image
C        NP                  (I*2)    Number of pixels in Output Image
C        KOMP                (I*2)    Background complement select (non-0)
C
C     /OUTLIN/    Output line assembly                      Length: 2000
C                     ANNOTE,TXTSET
C
C        LINE(4000)          (Byte)   Line array
C
C     /SYMBOL/    Symbol map table                          Length: 8640
C                     SYMBIN,LINES4,CLAN54
C
C        ISYMB(3,48,60)      (I*2)    3 words/line 48 lines/symbol 60 symbols
C                                      (bit map packed left-to-right in word)
C
C Orientation conventions:
C ----------------------
C        A CRT-based coordinate convention is used:
C                Origin is in the upper left corner
C                The pixel index advances to the right (pixel direction)
C                The line index advances downward (line direction)
C
C                         Pixel direction
C                             --->
C                        +------------+
C                        !    Top     !
C                        !L          R!
C                    !   !e  IMAGE   i!
C     Line direction !   !f  FRAME   g!
C                    v   !t          h!
C                        !           t!
C                        !   Bottom   !
C                        +------------+
C
C
C--------------------------------------------------------------------------
      COMMON /OUTIMG/ NL,NP,KOMP
      COMMON /INPIMG/ NIMG,FILNAM(10,2),JL(2),JP(2),KL(2),KP(2),
     *  INP(2),IL(2),LL(2),IP(2),LP(2),IC(2)
      BYTE FILNAM
      COMMON /OUTLIN/ LINE(4000)
      BYTE LINE
      COMMON /IMGASG/ LTMP(4000)
      BYTE LTMP
      COMMON /MODEU4/ IPRNT
      DIMENSION ISYM1(8),ISYM2(8)
      DATA ISYM1/1H ,1H.,1H-,1H+,1H*,1H0,1H8,1H</
      DATA ISYM2/1H ,1H ,1H ,1H ,1H ,1H0,1H8,1H>/
```

```
      CALL USER
      CALL SINDEX
      CALL SYMBIN
      CALL FILSET
      IF(IPRNT.EQ.0) TYPE 500,NL,NP
500   FORMAT(/,1X,'ASSEMBLING AND ANNOTATING IMAGE:',//,
     +  5X,I5,' LINES BY',I5,' PIXELS TO FILE "ANIMG.DAT"')
      IF(IPRNT.NE.0) TYPE 501,NL,NP
501   FORMAT(/,1X,'ASSEMBLING AND ANNOTATING IMAGE:',//,
     +  5X,I5,' LINES BY',I5,' PIXELS TO FILE "APRINT.DAT"')
      JCOL=128
      IF(NP.LT.128) JCOL=NP
      JCOL1=JCOL+1
      JSYMP=1H+
      JSYMM=1H-
      IF(IPRNT.NE.0) WRITE (4,510) (JSYMM,J=1,JCOL),JSYMP
510   FORMAT(1H1,'+',129A1)
      DO 400 L=1,NL
            DO 210 J=1,NP
210         LINE(J)=0
      IF(KOMP.EQ.0) GO TO 230
            DO 220 J=1,NP
220         LINE(J)="377
230   CONTINUE
      IF(NIMG.LE.0) GO TO 350
         DO 330 I=1,NIMG
         IF(L.LT.IL(I)) GO TO 330
         IF(L.GT.LL(I)) GO TO 330
         IU=I+1
         IN=INP(I)
         READ (IU,END=320) (LTMP(J),J=1,IN)
         JI=IP(I)
         JF=LP(I)
         JJ=JP(I)
         JC=IC(I)
            DO 310 J=JI,JF
            JJ=JJ+1
            IF(JC.NE.0) LTMP(JJ)=.NOT.LTMP(JJ)
            LINE(J)=LTMP(JJ)
310         CONTINUE
         GO TO 330
320               CONTINUE
                  CLOSE(UNIT=IU)
                  LL(I)=L-1
                  LI=L-IL(I)
                  KL(I)=LI
                  LI=LI+JL(I)
                  TYPE 600,I,LI,JL(I)
600               FORMAT(/,1X,'*** END OF FILE ON IMAGE',I2,
     +              ', ONLY',I5,' LINES (INCLUDING',I5,' SKIPS)')
330      CONTINUE
350   CONTINUE
      CALL TXTSET(L)
```

```
      IF(IPRNT.EQ.0) GO TO 390
      DO 380 J=1,JCOL
      LJ=LINE(J)
      IF(LJ.LT.0) LJ=LJ+256
      LJ=LJ/32+1
      LINE(J)=ISYM1(LJ)
      LTMP(J)=ISYM2(LJ)
380   CONTINUE
      LINE(JCOL1)=1H|
      WRITE (4,520) (LINE(J),J=1,JCOL1)
520   FORMAT(1X,'|',129A1)
      WRITE (4,530) (LTMP(J),J=1,JCOL)
530   FORMAT(1H+,1X,128A1)
      GO TO 400
390   WRITE (4) (LINE(J),J=1,NP)
400   CONTINUE
      IF(IPRNT.NE.0) WRITE (4,540) (JSYMN,J=1,JCOL),JSYMP
540   FORMAT(1X,'+',129A1)
      ENDFILE 4
      IF(NIMG.GE.1) CLOSE(UNIT=2)
      IF(NIMG.GE.2) CLOSE(UNIT=3)
      CLOSE(UNIT=4)
      END
```

```
      SUBROUTINE USER
C
C  Purpose: User interaction for image assembly
C                Output image specification
C                    Mode: raster image file or line-printer pseudo-image
C                    Frame size and background direct/complement select
C                Embedded image specification
C                    Image count (0 1 2)
C                    Image description
C                        File identification
C                        Input positioning (line & pixel skips)
C                        Size (line & pixel counts)
C                        Output location (starting line and pixel)
C                        Direct/complement select
C                Embedded annotation
C                    Message count (0-20)
C                    Message description
C                Message overlap assessment and editing
C
C  CALLed from: ANNOTE
C
C  CALLs: OVRLAP,UMSG
C
C-----------------------------------------------------------------------
      COMMON /OUTIMG/ NL,NP,KOMP
      COMMON /EMBIMG/ NIMG,FILNAM(10,2),JL(2),JP(2),KL(2),KP(2),
     +  INP(2),IL(2),LL(2),IP(2),LP(2),IC(2)
      BYTE FILNAM
      COMMON /EMBMSG/ NMSG,MSGO(20),MSGN(20),MSGM(20),M16(20),
     +  MSGL(20),MSGP(20),MSGC(20),TEXT(50,20)
      BYTE TEXT
      COMMON /MBOUND/ MIL(20),MLL(20),MIP(20),MLP(20)
      COMMON /MODEU4/ IPRNT
      DIMENSION NAMET(10)
      BYTE NAMET,NAME1
      EQUIVALENCE (NAME1,NAMET(1))
      DIMENSION LBLO(4),MSGT(50)
      BYTE LBLO,MSGT
      DATA NPIX,IPIX/4000,4000/
C-------------------------------------------------------- Defaults --+
      DATA IPRNT/0/                                                  !
      DATA NL,NP/800,800/                                            !
      DATA KOMP/0/                                                   !
      DATA NIMG/0/                                                   !
      DATA FILNAM/1HF,1HO,1HR,2*1HO,1H2,4*0,1HF,1HO,1HR,2*1HO,1H3,4*0/ !
      DATA JL,JP,KL,KP,IL,IP,IC/2*0,2*0,2*480,2*624,2*1,2*1,2*0/     !
      DATA INP,LL,LP/2*624,2*0,2*0/                                  !
      DATA NMSG/0/                                                   !
      DATA MSGO,MSGN,MSGM,MSGL,MSGP,MSGC/20*1,20*0,20*1,40*1,20*0/   !
      DATA TEXT/250*1H ,250*1H ,250*1H ,250*1H /                     !
C-------------------------------------------------------------------+
      DATA MIL,MLL,MIP,MLP/20*1,20*0,20*1,20*0/
      DATA LBLO/1HT,1HB,1HL,1HR/
```

```
      TYPE 500
500   FORMAT(/,1X,40(1H*),/,1X,'*',38X,'*',/,1X,
     +  '* IMAGE ASSEMBLY AND ANNOTATION MODULE *',
     +  /,1X,'*',38X,'*',/,1X,40(1H*))
C
C *** Output Image Specification
C
100   CONTINUE
      LIT=1HR
      IF(IPRNT.NE.0) LIT=1HP
      TYPE 510,LIT
510   FORMAT(//,3X,'OUTPUT IMAGE SPECIFICATION:',//,5X,
     +  'OUTPUT FILE MODE:',//,8X,
     +  'R  -  GRAY SCALE RASTER IMAGE FILE "ANIMG.DAT"',/,8X,
     +  'P  -  LINE PRINTER PSEUDO IMAGE FILE "APRINT.DAT"',//,
     +  5X,'MODE?  (R P)  ',A1)
      ACCEPT 1,LIT
      IF((LIT.EQ.1HR).OR.(LIT.EQ.1Hr)) IPRNT=0
      IF((LIT.EQ.1HP).OR.(LIT.EQ.1Hp)) IPRNT=1
      TYPE 520,NL
520   FORMAT(/,5X,'NUMBER OF LINES?',I5,5X,'(/ = NO CHNG)')
      ACCEPT *,NL
              NPIX=NP1X
              IF(IPRNT.NE.0) NPIX=128
              IF(NP.GT.NPIX) NP=NPIX
      TYPE 530,NPIX,NP
530   FORMAT(5X,'NUMBER OF PIXELS? (UP TO ',I4,')',I5,4X,'(/ = NO CHNG)')
      ACCEPT *,NP
              IF(NP.GT.NPIX) NP=NPIX
      LIT=1HN
      IF(KOMP.NE.0) LIT=1HY
      TYPE 535,LIT
535   FORMAT(5X,'COMPLEMENT BACKGROUND? (Y OR N)  ',A1)
      ACCEPT 1,LIT
      IF((LIT.EQ.1HN).OR.(LIT.EQ.1Hn)) KOMP=0
      IF((LIT.EQ.1HY).OR.(LIT.EQ.1Hy)) KOMP=1
C
C *** Embedded Image Specifications
C
      TYPE 540,NIMG
540   FORMAT(//,3X,'EMBEDDED IMAGES:',//,5X,
     +  'NUMBER OF IMAGES? (0 1 2)',I3,5X,'(/ = NO CHNG)')
      ACCEPT *,NIMG
              IF(NIMG.LT.0) NIMG=0
              IF(NIMG.GT.2) NIMG=2
      IF(NIMG.EQ.0) GO TO 200
      DO 150 I=1,NIMG
110   TYPE 550,I,(FILNAM(J,I),J=1,10)
550   FORMAT(5X,'IMAGE',I2,':',//,7X,
     +  'FILENAME? (UP TO 9 CHARACTERS)  ',10A1)
      ACCEPT 1,NAMET
1     FORMAT(50A1)
      IF(NAME1.EQ.1H ) GO TO 140
      IF(NAME1.EQ.1H/) GO TO 140
              IF((NAME1.GE.1HA).AND.(NAME1.LE.1HZ)) GO TO 120
```

```
                TYPE 560
560             FORMAT(1X,'*** FILENAME MUST START WITH LETTER')
                GO TO 110
120       DO 130 J=1,10
          FILNAM(J,I)=NAMET(J)
          IF(NAMET(J).LE.1H ) FILNAM(J,I)=0
          IF(NAMET(J).EQ.1H/) FILNAM(J,I)=0
          IF(J.EQ.1) GO TO 130
          IF(FILNAM(J-1,I).EQ.0) FILNAM(J,I)=0
130       CONTINUE
140       CONTINUE
          TYPE 570,JL(I)
570       FORMAT(7X,'SKIP LINES INTO INPUT IMAGE?',I5,5X,'(/ = NO CHNG)')
          ACCEPT *,JL(I)
          TYPE 580,IPIX,JP(I)
580       FORMAT(7X,'SKIP PIXELS INTO INPUT IMAGE? (<',I4,')',I5,5X,
     *      '(/ = NO CHNG)')
          ACCEPT *,JP(I)
                IF(JP(I).GE.IPIX) JP(I)=IPIX-1
                IF(KL(I).GT.NL) KL(I)=NL
          TYPE 590,NL,KL(I)
590       FORMAT(7X,'NUMBER OF LINES? (UP TO ',I4,')',I6,5X,'(/ = NO CHNG)')
          ACCEPT *,KL(I)
                IF(KL(I).GT.NL) KL(I)=NL
                MAXP=NP
                KTMP=IPIX-JP(I)
                IF(KTMP.LT.MAXP) MAXP=KTMP
                IF(KP(I).GT.MAXP) KP(I)=MAXP
          TYPE 600,MAXP,KP(I)
600       FORMAT(7X,'NUMBER OF PIXELS? (UP TP ',I4,')',I6,5X,'(/ = NO CHNG)')
          ACCEPT *,KP(I)
                IF(KP(I).GT.MAXP) KP(I)=MAXP
          INP(I)=JP(I)+KP(I)
                ILU=NL+1-KL(I)
          TYPE 610,ILU,IL(I)
610       FORMAT(7X,'STARTING LINE? (RANGE 1 -',I4,')',I6,5X,
     *      '(/ = NO CHNG)')
          ACCEPT *,IL(I)
                IF(IL(I).LT.1) IL(I)=1
                IF(IL(I).GT.ILU) IL(I)=ILU
          LL(I)=IL(I)+KL(I)-1
                IPU=NP+1-KP(I)
          TYPE 620,IPU,IP(I)
620       FORMAT(7X,'STARTING PIXEL? (RANGE 1 -',I4,')',I6,4X,
     *      '(/ = NO CHNG)')
          ACCEPT *,IP(I)
                IF(IP(I).LT.1) IP(I)=1
                IF(IP(I).GT.IPU) IP(I)=IPU
          LP(I)=IP(I)+KP(I)-1
          LIT=1HN
          IF(IC(I).NE.0) LIT=1HY
          TYPE 630,LIT
630       FORMAT(7X,'COMPLEMENT IMAGE? (Y OR N)  ',A1)
          ACCEPT 1,LIT
          IF((LIT.EQ.1HN).OR.(LIT.EQ.1Hn)) IC(I)=0
          IF((LIT.EQ.1HY).OR.(LIT.EQ.1Hy)) IC(I)=1
150       CONTINUE
```

```
C
C ***   Embedded Annotation Specifications
C
200     CONTINUE
        TYPE 640,NMSG
640     FORMAT(//,3X,'EMBEDDED ANNOTATION:',//,5X,
     *    'NUMBER OF MESSAGES? (0 -  20)',I4,5X,'(/ = NO CHNG)')
        ACCEPT *,NMSG
                IF(NMSG.LT.0) NMSG=0
                IF(NMSG.GT.20) NMSG=20
        IF(NMSG.EQ.0) GO TO 360
        DO 320 N=1,NMSG
        NT=MSGN(N)
        IF(NT.EQ.0) GO TO 310
        TYPE 650
650     FORMAT(//,1X,'MSG     LINE  PIX     TEXT')
        IT=MSGO(N)
        JC=IN
        IF(MSGC(N).NE.0) JC=INC
        TYPE 660,N,LOLO(IT),MSGN(N),MSGL(N),MSGP(N),JC,
     *    (TEXT(I,N),I=1,NT)
660     FORMAT(1X,I2,1X,A1,I2,'X',2I5,1X,A1,2X,50A1)
        IF(MLL(N).GT.NL) GO TO 310
        IF(MLP(N).GT.NP) GO TO 310
        TYPE 670,N
670     FORMAT(/,5X,'EDIT MESSAGE',I3,'? (Y OR N)  N')
        ACCEPT 1,LIT
        IF((LIT.NE.1HY).AND.(LIT.NE.1Hy)) GO TO 320
310     CALL UMSG(N)
320     CONTINUE
330     TYPE 650
        DO 350 N=1,NMSG
        NT=MSGN(N)
        IF(NT.GT.0) GO TO 340
        TYPE 660,N
        GO TO 350
340     IT=MSGO(N)
        JC=IN
        IF(MSGC(N).NE.0) JC=INC
        TYPE 660,N,LOLO(IT),MSGN(N),MSGL(N),MSGP(N),JC,
     *    (TEXT(I,N),I=1,NT)
350     CONTINUE
        TYPE 680,NMSG
680     FORMAT(/,3X,'MESSAGE TO CHANGE? (1 -',I2,')',5X,
     *    '(/ = NO FURTHER CHNG)')
        N=0
        ACCEPT *,N
        IF((N.LT.1).OR.(N.GT.NMSG)) GO TO 360
        CALL UMSG(N)
        GO TO 330
360     CONTINUE
        CALL OVRLAP
```

```
      TYPE 750
750   FORMAT(/,1X,'USER SPECIFICATION COMPLETE:',/,3X,30(1H*),//,5X,
     * 'REVIEW? (Y OR N)  N')
      ACCEPT 1,LIT
      IF(LIT.EQ.1HY) GO TO 100
      IF(LIT.EQ.1Hy) GO TO 100
      RETURN
      END
```

```
      SUBROUTINE UMSG(N)
C
C  Purpose: User interaction for image annotation - message N
C              Message description
C                  Orientation (Top Bottom Left Right)
C                  Character count
C                  Symbol size (1x 2x 3x 4x)
C                  Output location (center line & pixel)
C                  Direct/complement select
C                  Text
C
C  Called from: USER.OVRLAP
C
C----------------------------------------------------------------------
      COMMON /OUTIMG/ NL,NP,NBMP
      COMMON /IMAGE/ NMSG,MSGO(20),MSGN(20),MSGN(20),MSIG(20),
     *  MSGL(20),MSGP(20),MSGC(20),TEXT(50,20)
      BYTE TEXT
      COMMON /MSGPOS/ MJL(20),MLL(20),MJP(20),MLP(20)
      DIMENSION LOLO(4),NCHAR(4),MSGT(50)
      BYTE LOLO,MSGT
      DIMENSION NSEP(256)
      BYTE NSEP
      DATA LOLO/1HT,1HB,1HL,1HR/
      DATA NSEP/[illegible]
     *  [illegible]
      ITEMP=MSGO(N)
      TYPE 800,N,LOLO(IT)
800   FORMAT(/,5X,'MESSAGE',I3,':',//,7X,
     *  'ORIENTATION IN FRAME, TOP OF SYMBOL TOWARDS',/,30X,
     *  'T = TOP',/,30X,'B = BOTTOM',/,30X,'L = LEFT',/,30X,
     *  'R = RIGHT',//,9X,'ORIENTATION? (T B L R)   ',A1)
      ACCEPT 1,LIT
1     FORMAT(50A1)
      IF((LIT.EQ.1HT).OR.(LIT.EQ.1HT)) MSGO(N)=1
      IF((LIT.EQ.1HB).OR.(LIT.EQ.1HB)) MSGO(N)=2
      IF((LIT.EQ.1HL).OR.(LIT.EQ.1HL)) MSGO(N)=3
      IF((LIT.EQ.1HR).OR.(LIT.EQ.1HR)) MSGO(N)=4
      ITEMP=MSGO(N)
             DO 210 NI=1,4
             NI10=10*NI
             NT=50
             JTEMP/NI10
             IF(IT.GE.3) JTEMP/NI10
             IF((NL.LT.NI10).OR.(NP.LT.NI10)) JT=0
             IF(NT.GT.JT) NT=JT
210          NCHAR(NI)=NT
             IF(MSGN(N).GT.NCHAR(1)) MSGN(N)=NCHAR(1)
      TYPE 810,N,(NCHAR(NI),NI,NI=1,4),MSGN(N)
810   FORMAT(7X,'MESSAGE',I3,' LENGTH',4(/,11X,'0 -',I3,
     *  ' CHARACTERS AT',I2,'X'),//,9X,'CHARACTER COUNT?',I4,5X,
     *  '(/ = NO CHANGE, 0 = DELETE)')
      ACCEPT *,MSGN(N)
             IF(MSGN(N).LT.0) MSGN(N)=0
             IF(MSGN(N).GT.NCHAR(1)) MSGN(N)=NCHAR(1)
      NTEMP=MSGN(N)
```

```
      [illegible](N)=1
      [illegible](N)=0
      [illegible](N)=1
      [illegible](N)=0
      IF(NT.EQ.0) GO TO 200
            [illegible]=1
            [illegible]=2
            [illegible]=3
            [illegible]=4
            IF(NT.GT.[illegible](2)) [illegible]
            IF(NT.GT.[illegible](3)) [illegible]
            IF(NT.GT.[illegible](4)) [illegible]
            [illegible](N)
220         IF(NT.LE.[illegible]) GO TO 230
            [illegible]=1
            GO TO 220
230         [illegible](N)=NT
      TYPE 620,N,[illegible],[illegible],[illegible],[illegible],[illegible](N)
620   FORMAT([illegible],'[illegible]',I3,' SYMBOL SIZES (',A1,I4,A1,I4,A1,I4,A1,
     * ')',I3,[illegible],'(/ 0 NO CHANGE)')
      ACCEPT *,NT
            IF(NT.LT.1) NT=1
            IF(NT.GT.4) NT=4
240         IF(NT.LE.[illegible]) GO TO 250
            [illegible]=1
            GO TO 400
250   [illegible](N)=NT
            [illegible]=NT
      [illegible](N)=[illegible]
            [illegible]
            [illegible]/2
            [illegible]/2
            [illegible]
            IF(I1.GE.3) [illegible]
            [illegible]
            [illegible]
            [illegible](N)
            IF(L.LT.[illegible]) L=[illegible]
            IF(L.GT.[illegible]) L=[illegible]
            [illegible](N)=L
      TYPE 630,[illegible],[illegible],[illegible](N)
630   FORMAT([illegible],'[illegible] CENTER AT LINES (',I4,' -',I5,')',I4,[illegible],
     * '(/ 0 NO CHANGE)')
      ACCEPT *,L
            IF(L.LT.[illegible]) L=[illegible]
            IF(L.GT.[illegible]) L=[illegible]
      [illegible](N)=L
      [illegible](N)=[illegible]
      [illegible](N)=[illegible]
            [illegible]
            IF(I1.GE.3) [illegible]
            [illegible]
            [illegible]
```

```
            [illegible]
            IF([illegible]) [illegible]
            IF([illegible]) [illegible]
            [illegible]
      TYPE [illegible]
840   FORMAT([illegible] CENTER [illegible]
     *  [illegible]
      ACCEPT [illegible]
            IF([illegible]) [illegible]
            IF([illegible]) [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      IF([illegible]) [illegible]
      TYPE [illegible]
850   FORMAT([illegible]
      ACCEPT [illegible]
      IF([illegible]) [illegible]
      IF([illegible]) [illegible]
      TYPE [illegible]
860   FORMAT([illegible]
     *  [illegible]
     *  [illegible]
     *  [illegible]
     *  [illegible]
     *  [illegible]
      TYPE 870
870   FORMAT([illegible]
     *  [illegible]
     *  [illegible]
     *  [illegible]
     *  [illegible]
     *  [illegible]
     *  [illegible]
     *  [illegible]
      TYPE [illegible]
880   FORMAT([illegible] TEXT [illegible]
     *  [illegible]
      [illegible]
      TYPE [illegible]
890   FORMAT([illegible]
      ACCEPT [illegible]
      DO 200 [illegible]
      IF([illegible]) GO TO 270
      [illegible]
200   CONTINUE
      GO TO 300
270   IF([illegible]) GO TO 300
280   DO 290 [illegible]
            [illegible]
            IF([illegible]) [illegible]
      TEXT([illegible]
      IF([illegible]) TEXT([illegible]
290   CONTINUE
300   RETURN
      END
```

```
      SUBROUTINE OVRLAP
C
C  Purpose: Assess annotation conflicts
C               Option to edit overwritten message
C               Option to edit overwriting message
C
C  Called from:  [illegible]
C
C  Calls: UMSG
C
C-----------------------------------------------------------------------
      COMMON /[illegible]/ [illegible]
      COMMON /[illegible]/ [illegible](10,7),[illegible](2),[illegible](2),[illegible](2),[illegible](2),
     *  [illegible](2),[illegible](2),[illegible](2),[illegible](2),[illegible](2),[illegible](2)
      BYTE FILNAM
      COMMON /[illegible]/ NMSG,[illegible](30),[illegible](30),[illegible](30),[illegible](30),
     *  [illegible](30),[illegible](30),[illegible](30),TEXT(50,30)
      BYTE TEXT
      COMMON /[illegible]/ [illegible](30),[illegible](30),[illegible](30),[illegible](30)
      IF(NMSG.LE.1) GO TO 640
      [illegible]=1
600   DO 620 [illegible]=1,NMSG
      [illegible]([illegible])
      IF([illegible].LE.0) GO TO 620
      [illegible]=[illegible]+1
      DO 610 [illegible]=[illegible],NMSG
      [illegible]([illegible])
      IF([illegible].LE.0) GO TO 610
      IF([illegible]([illegible]).GT.[illegible]([illegible])) GO TO 610
      IF([illegible]([illegible]).GT.[illegible]([illegible])) GO TO 610
      IF([illegible]([illegible]).GT.[illegible]([illegible])) GO TO 610
      IF([illegible]([illegible]).GT.[illegible]([illegible])) GO TO 610
      TYPE 910,[illegible],[illegible]
910   FORMAT(1X,'MESSAGE',I3,' OVERLAPS MESSAGE',I3,//,1X,
     *  'MSG  LINES     PIXELS  TEXT')
      TYPE 920,[illegible],[illegible],[illegible],[illegible],[illegible],(TEXT(I,[illegible]),I=1,[illegible])
      TYPE 920,[illegible],[illegible],[illegible],[illegible],[illegible],(TEXT(I,[illegible]),I=1,[illegible])
920   FORMAT(1X,I4,I5,'-',I4,I5,'-',I4,1X,50A1)
      TYPE 930,[illegible],[illegible]
930   FORMAT(/,1X,'EDIT ENTRY',I3,' OR',I3,'? (Y OR N)  N')
      ACCEPT 1,LIT
1     FORMAT(A1)
      IF((LIT.EQ.'N').OR.(LIT.EQ.'n')) GO TO 610
610   CONTINUE
620   CONTINUE
      GO TO 640
630   TYPE 940,[illegible]
940   FORMAT(1X,'EDIT',I3,'? (Y OR N)  N')
      ACCEPT 1,LIT
      IF((LIT.EQ.'Y').OR.(LIT.EQ.'y')) CALL UMSG([illegible])
      TYPE 940,[illegible]
      ACCEPT 1,LIT
      IF((LIT.EQ.'Y').OR.(LIT.EQ.'y')) CALL UMSG([illegible])
      GO TO 600
640   RETURN
      END
```

```
      SUBROUTINE SINDEX
C
C  Purpose: Convert message texts from ASCII to Symbol Map Indices
C
C  Called from: ANNOTE
C
C-----------------------------------------------------------------------
      COMMON /[illegible]/ NMSG,MSGX(20),MSGY(20),MSGN(20),[illegible](20),
     *   MSGL(20),MSGP(20),MSGC(20),TEXT(50,20)
      BYTE TEXT
      DIMENSION ISYMB(256)
      DATA ISYMB/33*64,44,6*64,40,7*64,41,42,39,37,45,38,46,48,
     *   [illegible],
     *   1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,
     *   23,24,25,26,6*64,57,58,59,60,53,51,52,[illegible],
     *   130*64/
      IF(NMSG.EQ.0) GO TO 130
      DO 120 M=1,NMSG
      NT=MSGN(M)
      IF(NT.EQ.0) GO TO 120
            DO 110 N=1,NT
            NA=TEXT(N,M)
            IF(NA.LE.0) NA=32
            TEXT(N,M)=ISYMB(NA)
110         CONTINUE
120   CONTINUE
130   RETURN
      END
```

```
      SUBROUTINE SYMBIN
C
C  Purpose: Input Symbol Bit-map tables from file SYMBOL.DAT
C
C  Called from: ANNOTE
C
C-----------------------------------------------------------------------
      COMMON /SYMBOL/ ISYMB(3,48,60)
      DIMENSION JSYMB(8640)
      EQUIVALENCE (ISYMB(1,1,1),JSYMB(1))
              DIMENSION JJSYMB(17280)
              BYTE JJSYMB
              EQUIVALENCE (JJSYMB,JSYMB)
      OPEN(UNIT=1,TYPE='OLD',NAME='SYMBOL',FORM='UNFORMATTED',
     *  READONLY)
      JI=1
      JL=1440
      DO 110 I=1,6
      READ (1) (JSYMB(J),J=JI,JL)
      JI=JL+1
      JL=JL+1440
110   CONTINUE
      CLOSE(UNIT=1)
      RETURN
      END
```

```
      SUBROUTINE FILSET
C
C  Purpose: Open and position image files
C
C  Called from: ANNOTE
C
C-----------------------------------------------------------------------
      COMMON /ENDING/ NIMG,FILNAM(10,2),JL(2),JP(2),KL(2),KP(2),
     *   INP(2),IL(2),LL(2),IP(2),LP(2),IC(2)
      BYTE FILNAM
      COMMON /MODEU4/ IPRNT
      DIMENSION F1(10),F2(10)
      BYTE F1,F2
      EQUIVALENCE (F1(1),FILNAM(1,1)),(F2(1),FILNAM(1,2))
      IF(NIMG.GE.1) OPEN(UNIT=2,TYPE='OLD',NAME=F1,FORM='UNFORMATTED',
     *   READONLY)
      IF(NIMG.GE.2) OPEN(UNIT=3,TYPE='OLD',NAME=F2,FORM='UNFORMATTED',
     *   READONLY)
      IF(NIMG.LE.0) GO TO 100
      DO 130 I=1,NIMG
      JL1=JL(I)
      IF(JL1.LE.0) GO TO 130
      IU=I+1
          DO 110 J=1,JL1
          READ (IU,END=120,ERR=120)
110       CONTINUE
      GO TO 130
120           TYPE 500,I,J
500           FORMAT(/,1X,'*** EOF/ERR IMAGE',I2,' AT SKIP LINE',I5)
              STOP
130   CONTINUE
100   CONTINUE
      IF(IPRNT.NE.0) GO TO 150
      OPEN(UNIT=4,TYPE='NEW',NAME='ABIMG',FORM='UNFORMATTED')
      GO TO 160
150   OPEN(UNIT=4,TYPE='NEW',NAME='APRINT')
160   RETURN
      END
```

```
      SUBROUTINE TXTSET(L)
C
C  Purpose: Set annotation line segments for all messages active on line L
C               Loop on messages
C                   Branch on orientation (T,B vs L,R)
C                       Identify Symbol Map lines or columns to extract
C                       Extract Symbol Map lines or columns
C                       Resample lines or columns to required symbol size
C                   Transfer message segment to output line buffer
C
C  CALLed from: ANNOTE
C
C  CALLS: INDEX4,LINES4,CLMNS4,RSMPL1,RSMPL2,RSMPL3,RSMPL4
C
C-----------------------------------------------------------------
      COMMON /EMBMSG/ NMSG,MSGO(20),MSGN(20),MSGM(20),M16(20),
     *  MSGL(20),MSGP(20),MSGC(20),TEXT(50,20)
      BYTE TEXT
      COMMON /MBOUND/ MIL(20),MLL(20),MIP(20),MLP(20)
      COMMON /OUTLIN/ LINE(4000)
      BYTE LINE
      COMMON /IMGMSG/ LTMP(4000)
      BYTE LTMP
      COMMON /LCLMSG/ MT,ML,ML4,ML4(2),KT,L4T(3,2),CHAR(64)
      BYTE CHAR
      IF(NMSG.EQ.0) GO TO 510
      DO 500 M=1,NMSG
      NT=MSGN(M)
      IF(NT.LE.0) GO TO 500
      LFI=L-MIL(M)
      IF(LFI.LT.0) GO TO 500
      LFL=MLL(M)-L
      IF(LFL.LT.0) GO TO 500
      IT=MSGO(M)
      MT=MSGM(M)
      MX=M16(M)
      JI=MIP(M)
      JL=MLP(M)
      DO 110 J=JI,JL
110   LTMP(J)=0
      IF(IT.GE.3) GO TO 300
          ML=LFI+1
          IF(IT.EQ.2) ML=LFL+1
          CALL INDEX4
          DO 260 K=1,NT
              DO 210 J=1,MX
210           CHAR(J)=0
          IF(ML4.EQ.0) GO TO 220
          KT=TEXT(K,M)
          CALL LINES4
          IF(MT.EQ.1) CALL RSMPL1
          IF(MT.EQ.2) CALL RSMPL2
          IF(MT.EQ.3) CALL RSMPL3
          IF(MT.EQ.4) CALL RSMPL4
```

```
220             KX=(K-1)*MX
                IF(IT.EQ.2) GO TO 230
                IPT=JI+KX
                INC=1
                GO TO 240
230             IPT=JL-KX
                INC=-1
240                     DO 250 J=1,MX
                        LTMP(IPT)=CHAR(J)
250                     IPT=IPT+INC
260             CONTINUE
        GO TO 400
300     CONTINUE
                NC=LFL
                IF(IT.EQ.4) NC=LFI
                K=NC/MX
                NL=NC+1-K*MX
                K=K+1
                CALL INDEX4
                        DO 310 J=1,MX
310                     CHAR(J)=0
                IF(NL4.EQ.0) GO TO 320
                KT=TEXT(K,M)
                CALL CLMNS4
                IF(MT.EQ.1) CALL RSMPL1
                IF(MT.EQ.2) CALL RSMPL2
                IF(MT.EQ.3) CALL RSMPL3
                IF(MT.EQ.4) CALL RSMPL4
320             IF(IT.EQ.4) GO TO 330
                IPT=JI
                INC=1
                GO TO 340
330             IPT=JL
                INC=-1
340                     DO 350 J=1,MX
                        LTMP(IPT)=CHAR(J)
350                     IPT=IPT+INC
400     CONTINUE
                IF(MSGC(M).EQ.0) GO TO 420
                        DO 410 J=JI,JL
410                     LTMP(J)=.NOT.LTMP(J)
420             DO 430 J=JI,JL
430             LINE(J)=LTMP(J)
500     CONTINUE
510     RETURN
        END
```

```
      SUBROUTINE INDEX4
C
C  Purpose: Determine Symbol Map lines or columns to be extracted
C                N4  -  Number of lines/columns for given
C                         message line, symbol size
C                L4  -  Index of first line/column to be extracted
C                         from Symbol Map
C
C  CALLed from: TXTSET
C
C------------------------------------------------------------------------
      COMMON /LCLMSG/ MT,ML,NL4,ML4(2),KT,L4T(3,2),CHAR(64)
      BYTE CHAR
      DIMENSION N4(64,4),L4(64,4)
      BYTE N4,L4
      DATA N4/2*0,12*2,2*0,48*0,  4*0,24*2,4*0,32*0,
     +  6*0,1,2,1,1,2,1,1,2,1,1,2,1,1,2,1,1,2,1,1,2,1,1,2,1,1,2,1,
     +  1,2,1,1,2,1,1,2,1,6*0,16*0,  8*0,48*1,8*0/
      DATA L4/0,0,2,6,10,14,18,22,26,30,34,38,42,46,0,0,48*0,
     +  4*0,1,3,5,7,9,11,13,15,17,19,21,23,25,27,29,31,33,35,37,39,
     +  41,43,45,47,4*0,32*0,  6*0,1,2,4,5,6,8,9,10,12,13,14,16,17,18,
     +  20,21,22,24,25,26,28,29,30,32,33,34,36,37,38,40,41,42,
     +  44,45,46,48,6*0,16*0,  8*0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,
     +  15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,
     +  35,36,37,38,39,40,41,42,43,44,45,46,47,48,8*0/
      NL4=N4(ML,MT)
      IF(NL4.EQ.0) GO TO 120
      KL4=L4(ML,MT)-1
      DO 110 J=1,NL4
110   ML4(J)=KL4+J
120   RETURN
      END
```

```
      SUBROUTINE LINES4
C
C  Purpose: Extract line(s) from Symbol Map (3 words/line)
C
C  CALLed from: TXTSET
C
C-----------------------------------------------------------------------
      COMMON /LCLMSG/ MT,NL,NL4,NL4(2),KT,L4T(3,2),CHAR(64)
      BYTE CHAR
      COMMON /SYMBOL/ ISYMB(3,48,60)
      DO 120 J=1,NL4
      KL4=NL4(J)
            DO 110 JJ=1,3
110         L4T(JJ,J)=ISYMB(JJ,KL4,KT)
120   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CLMNS4
C
C  Purpose: Extract column(s) from Symbol Map (3 words/column)
C
C  CALLed from: TXTSET
C
C-----------------------------------------------------------------
      COMMON /LCLMSG/ MT,ML,NL4,ML4(2),KT,L4T(3,2),CHAR(64)
      BYTE CHAR
      COMMON /SYMBOL/ ISYMB(3,48,60)
      DIMENSION JJT(48),JKT(48)
      BYTE JJT,JKT
      DATA JJT/16*1,16*2,16*3/
      DATA JKT/-15,-14,-13,-12,-11,-10,-9,-8,-7,-6,-5,-4,-3,-2,-1,0,
     +         -15,-14,-13,-12,-11,-10,-9,-8,-7,-6,-5,-4,-3,-2,-1,0,
     +         -15,-14,-13,-12,-11,-10,-9,-8,-7,-6,-5,-4,-3,-2,-1,0/
      DO 130 J=1,NL4
      JL4=ML4(J)
      JJW=JJT(JL4)
      JJS=JKT(JL4)
      KL4=0
            DO 120 JJ=1,3
            NW=0
                  DO 110 JJP=1,16
                  KL4=KL4+1
                  ITS=ISYMB(JJW,KL4,KT)
                  JTS=JJS
                  NB=IISHFT(ITS,JTS).AND."1
                  ITS=NB
                  JTS=16-JJP
                  NW=NW.OR.IISHFT(ITS,JTS)
110               CONTINUE
            L4T(JJ,J)=NW
120         CONTINUE
130   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE RSMPL1
C
C  Purpose: Resample 4x Symbol Map to 1x symbol size
C
C                          A B C D
C      Symbol Map          E F G H      Resample   Z = (F.or.K).and.(G.or.J)
C       4x4 cell           I J K L       Pixel
C                          M N O P
C
C  CALLed from: TXTSET
C
C----------------------------------------------------------------------
      COMMON /LCLMSG/ MT,ML,ML4,ML4(2),KT,L4T(3,2),CHAR(64)
      BYTE CHAR
      JCHR=2
      DO 120 JJ=1,3
      L4T1=L4T(JJ,1)
      L4T2=L4T(JJ,2)
      ITS=L4T2
      JTS=1
      L4F=L4T1.OR.IISHFT(ITS,JTS)
      ITS=L4T1
      L4F=L4F.AND.(IISHFT(ITS,JTS).OR.L4T2)
      JS=-14
            DO 110 JP=1,4
            JCHR=JCHR+1
            ITS=L4F
            JTS=JS
            NB=IISHFT(ITS,JTS).AND."1
            IF(NB.NE.0) CHAR(JCHR)="377
110         JS=JS+4
120   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE RSMPL2
C
C  Purpose: Resample 4x Symbol Map to 2x symbol size
C
C                         A B C D                          W = B.and.E
C       Symbol Map        E F G H       Resample   WX      X = C.and.H
C        4x4 cell         I J K L        Pixels    YZ      Y = I.and.N
C                         M N O P                          Z = L.and.O
C
C  CALLed from: TXTSET
C
C-----------------------------------------------------------------------
      COMMON /LCLMSG/ NT,NL,NL4,ML4(2),KT,L4T(3,2),CHAR(64)
      BYTE CHAR
      IF((NL.AND."1).EQ.0) GO TO 110
      MSK1="104210
      MSK2="21042
      GO TO 120
110   MSK1="21042
      MSK2="104210
120   JCHR=4
      DO 140 JJ=1,3
      L4T1=L4T(JJ,1)
      L4T2=L4T(JJ,2)
      ITS=L4T1
      JTS=1
      L4S1=IISHFT(ITS,JTS).AND.L4T2
      ITS=L4T2
      L4S2=L4T1.AND.IISHFT(ITS,JTS)
      L4F=(MSK1.AND.L4S1).OR.(MSK2.AND.L4S2)
      JS=-15
              DO 130 JP=1,8
              JCHR=JCHR+1
              ITS=L4F
              JTS=JS
              NB=IISHFT(ITS,JTS).AND."1
              IF(NB.NE.0) CHAR(JCHR)="377
130           JS=JS+2
140   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE RSMPL3
C
C  Purpose: Resample 4x Symbol map to 3x symbol size
C
C                       A B C D                      R = A
C      Symbol Map       E F G H     Resample  RST    S = B.or.C
C       4x4 cell        I J K L      Pixels   UVW    T = D
C                       M N O P               XYZ    U = E.or.I
C                                                    V = (F.or.K).and.(G.or.J)
C                                                    W = H.or.L
C                                                    X = M
C                                                    Y = N.or.O
C                                                    Z = P
C
C  CALLed from: TXTSET
C
C---------------------------------------------------------------------
      COMMON /LCLMSG/ NT,NL,NL4,ML4(2),KT,L4T(3,2),CHAR(64)
      BYTE CHAR
      DIMENSION JST(12)
      DATA JST/-15,-14,-12,-11,-10,-8,-7,-6,-4,-3,-2,0/
      JCHR=0
      DO 140 JJ=1,3
      L4T1=L4T(JJ,1)
      IF(NL4.NE.1) GO TO 110
         ITS=L4T1
         JTS=1
         L4F=L4T1.OR.(IISHFT(ITS,JTS).AND."42104)
         GO TO 120
110   L4T2=L4T(JJ,2)
         L4O=L4T1.OR.L4T2
         ITS=L4T2
         JTS=1
         L4A=L4T1.OR.IISHFT(ITS,JTS)
         ITS=L4T1
         L4A=L4A.AND.(IISHFT(ITS,JTS).OR.L4T2)
         L4F=(L4O.AND."114631).OR.(L4A.AND."42104)
120         DO 130 JP=1,12
            JCHR=JCHR+1
            JS=JST(JP)
            ITS=L4F
            JTS=JS
            NB=IISHFT(ITS,JTS).AND."1
            IF(NB.NE.0) CHAR(JCHR)="377
130         CONTINUE
140   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE RSMPL4
C
C  Purpose: Resample 4x Symbol map to 4x symbol size
C              Direct mapping
C
C  CALLed from: TXTSET
C
C-----------------------------------------------------------------
      COMMON /LCLMSG/ NT,NL,NL4,NL4(2),NT,LAT(3,2),CHAR(64)
      BYTE CHAR
      JCHR=0
      DO 120 JJ=1,3
      LAF=LAT(JJ,1)
      JS=-15
          DO 110 JP=1,16
          JCHR=JCHR+1
          ITS=LAF
          JTS=JS
          ND=IISHFT(ITS,JTS).AND."1
          IF(ND.NE.0) CHAR(JCHR)="377
110       JS=JS+1
120   CONTINUE
      RETURN
      END
```